



**Aide à la conception des ensembles
machine-convertisseur-commande : apport d'une démarche
système expert**
Laurent Gerbaud

► **To cite this version:**

Laurent Gerbaud. Aide à la conception des ensembles machine-convertisseur-commande : apport d'une démarche système expert. Energie électrique. Institut National Polytechnique Grenoble (INPG), 1993. Français. $\langle$ NNT : $\rangle$. $\langle$ tel-02009245 $\rangle$

HAL Id: tel-02009245
<https://hal.science/tel-02009245v1>

Submitted on 6 Feb 2019

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



HAL Authorization



THESE

présentée par

Laurent GERBAUD

Ingénieur E.N.S.I.E.G.

pour obtenir le grade de DOCTEUR

de l'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

(Arrêté ministériel du 30 mars 1992)

(Spécialité génie électrique)

=====

AIDE A LA CONCEPTION DES ENSEMBLES MACHINE - CONVERTISSEUR - COMMANDE : APPORT D'UNE DEMARCHE SYSTEME EXPERT

=====

Date de soutenance : 23 Avril 1993

Composition du Jury :

Monsieur

R. PERRET

Président

Messieurs

N. GIAMBIASI

Rapporteur

C. SOL

Rapporteur

Y. BAUDON

J. BIGEON

G. CHAMPENOIS

Thèse préparée au sein du Laboratoire d'Electrotechnique de Grenoble

THESE

présentée par

Laurent GERBAUD

Ingénieur E.N.S.I.E.G.

pour obtenir le grade de DOCTEUR

de l'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

(Arrêté ministériel du 30 mars 1992)

(Spécialité génie électrique)

=====

**AIDE A LA CONCEPTION DES
ENSEMBLES MACHINE - CONVERTISSEUR - COMMANDE :
APPORT D'UNE DEMARCHE SYSTEME EXPERT**

=====

Date de soutenance : 23 Avril 1993

Composition du Jury :

Monsieur	R. PERRET	Président
Messieurs	N. GIAMBIASI	Rapporteur
	C. SOL	Rapporteur
	Y. BAUDON	
	J. BIGEON	
	G. CHAMPENOIS	

Thèse préparée au sein du Laboratoire d'Electrotechnique de Grenoble

PRESIDENT DE L'INSTITUT
Monsieur Maurice RENAUD

Année 1991/1992

PROFESSEURS DES UNIVERSITES

BARIBAUD	Michel	ENSERG
BARRAUD	Alain	ENSIEG
BAUDELET	Bernard	ENSPG
BAUDIN	Gérard	UFR PGP
BEAUFILS	Jean-Pierre	ENSIEG/ILL
BOIS	Philippe	ENSHMG
BONNET	Guy	ENSPG
BOUVIER	Gérard	ENSERG
BRISSONNEAU	Pierre	ENSIEG
BRUNET	Yves	CUEFA
CAILLERIE	Denis	ENSHMG
CAVAIGNAC	Jean-François	ENSPG
CHARTIER	Germain	ENSPG
CHENEVIER	Pierre	ENSERG
CHERADAME	Hervé	UFR/PGP
CHERUY	Arlette	ENSIEG
CHOYET	Alain	ENSERG
COGNET	Gérard	ENSHMG
COLINET	Catherine	ENSEEG
COMMAULT	Christian	ENSIEG
CORNUT	Bruno	ENSIEG
COULOMB	Jean-Louis	ENSIEG
CROWLEY	James	ENSIMAG
DALARD	Francis	ENSEEG
DARVE	Félix	ENSHMG
DELLA DORA	Jean	ENSIMAG
DEPEY	Maurice	ENSERG
DEPORTES	Jacques	ENSPG
DEROO	Daniel	ENSEEG
DESRE	Pierre	ENSEEG
DIARD	Jean-Paul	ENSEEG
DOLMAZON	Jean-Marc	ENSERG
DURAND	Francis	ENSEEG
DURAND	Jean-Louis	ENSPG
FAUTRELLE	Yves	ENSHMG
FOGGIA	Albert	ENSIEG
FONLUPT	Jean	ENSIMAG
FOULARD	Claude	ENSIEG
GALERIE	Alain	ENSEEG
GANDINI	Alessandro	UFR/PGP
GAUBERT	Claude	ENSPG
GENTIL	Pierre	ENSERG
GENTIL	Sylviane	ENSIEG
GREVEN	Hélène	CUEFA
GUERIN	Bernard	ENSERG
GUYOT	Pierre	ENSEEG
IVANES	Marcel	ENSIEG
JALLUT	Christian	ENSEEG

JANOT	Marie-Thérèse	ENSERG
JAUSSAUD	Pierre	ENSIEG
JOST	Rémy	ENSPG
JOUBERT	Jean-Claude	ENSPG
JOURDAIN	Geneviève	ENSIEG
KUENY	Jean-Louis	ENSHMG
LACHENAL	Dominique	UFR PGP
LACQUME	Jean-Louis	ENSIEG
LADET	Pierre	ENSIEG
LESIEUR	Marcel	ENSHMG
LESPINARD	Georges	ENSHMG
LIENARD	Joël	ENSIEG
LONGUEUEUE	Jean-Pierre	ENSPG
LORET	Benjamin	ENSHMG
LOUCHET	François	ENSEEG
LUCAZEAU	Guy	ENSEEG
LUX	Augustin	ENSIMAG
MASSE	Philippe	ENSIEG
MASSELOT	Christian	ENSIEG
MAZARE	Guy	ENSIMAG
MICHEL	Gérard	ENSIMAG
MOHR	Roger	ENSIMAG
MOREAU	René	ENSHMG
MORET	Roger	ENSIEG
MOSSIÈRE	Jacques	ENSIMAG
OBLED	Charles	ENSHMG
OZIL	Patrick	ENSEEG
PANANAKAKIS	Georges	ENSERG
PAULEAU	Yves	ENSEEG
PERRET	Robert	ENSIEG
PIAU	Jean-Michel	ENSHMG
PIC	Etienne	ENSERG
PLATEAU	Brigitte	ENSIMAG
POUPOT	Christian	ENSERG
RAMEAU	Jean-Jacques	ENSEEG
REINISCH	Raymond	ENSPG
RENAUD	Maurice	UFR/PGP
ROBERT	François	ENSIMAG
ROSSIGNOL	Michel	ENSPG
ROYE	Daniel	ENSIEG
SABONNADIÈRE	Jean-Claude	ENSIEG
SAGUET	Pierre	ENSERG
SAUCIER	Gabrièle	ENSIMAG
SCHLENKER	Claire	ENSPG
SCHLENKER	Michel	ENSPG
SILVY	Jacques	UFR/PGP
SIRIEYS	Pierre	ENSHMG
SOHM	Jean-Claude	ENSEEG
SOLER	Jean-Louis	ENSIMAG
SOUQUET	Jean-Louis	ENSEEG
TICHKIEWITCH	Serge	ENSHMG
TROMPETTE	Philippe	ENSHMG
TRYSTRAM	Denis	ENSGI
VEILLON	Gérard	ENSIMAG
VERJUS	Jean-Pierre	ENSIMAG
VINCENT	Henri	ENSPG
ZADWORNÝ	François	ENSERG

SITUATION PARTICULIERE

PROFESSEURS D'UNIVERSITE

DETACHEMENT

ENSERG	BLIMAN	Samuel	Mutation	
ENSPG	BLOCH	Daniel	Recteur	21/12/93
ENSIMAG	LATOMBE	Jean-Claude	Détachement	01/05/93
ENSHMG	PIERRARD	Jean-Marie	Disponible	

RETRAITE

ENSEEG	BONNETAIN	Lucien
--------	-----------	--------

DIRECTEURS DE RECHERCHE CNRS

ABELLO	Louis	MEUNIER	Gérard
ALDEBERT	Pierre	MICHEL	Jean-Marie
ALEMANY	Antoine	NAYROLLES	Bernard
ALLIBERT	Colette	PASTUREL	Alain
ALLIBERT	Michel	PEUZIN	Jean-Claude
ANSARA	Ibrahim	PHAM	Antoine
ARMAND	Michel	PIAU	Monique
AUDIER	Marc	PIQUE	Jean-Paul
AUGOYARD	Jean-François	POINSIGNON	Christiane
AVIGNON	Michel	PREJEAN	Jean-Jacques
BERNARD	Claude	RENOUARD	Dominique
BINDER	Gilbert	SENATEUR	Jean-Pierre
BLAISING	Jean-Jacques	SIFAKIS	Joseph
BONNET	Roland	SIMON	Jean-Paul
BORNARD	Guy	SUERY	Michel
BOUCHERLE	Jean-Xavier	TEODOSIU	Christian
CAILLET	Marcel	VACHAUD	Georges
CARRE	René	YAUCLIN	Michel
CHASSERY	Jean-Marc	WACK	Bernard
CHATILLON	Christian	YAYARI	Ali-Reza
CIBERT	Joël	YONNET	Jean-Paul
CLERMONT	Jean-Robert		
COURTOIS	Bernard		
CRIQUI	Patrick		
CRISTOLOYEANU	Sorin		
DAVID	René		
DION	Jean-Michel		
DOUSSIERE	Jacques		
DRIOLE	Jean		
DUCHET	Pierre		
DUGARD	Luc		
DURAND	Robert		
ESCUDIER	Pierre		
EUSTATHOPOULOS	Nicolas		
FINON	Dominique		
FRUCHARD	Robert		
GARNIER	Marcel		
GIROD	Jacques		
GLANGEAUD	François		
GUELIN	Pierre		
HOPFINGER	Emil		
JORRAND	Philippe		
JOUD	Jean-Charles		
KAMARINOS	Georges		
KLEITZ	Michel		
KOFMAN	Walter		
LACROIX	Claudine		
LANDAU	Ioan		
LAULHERE	Jean-Pierre		
LEGRAND	Michel		
LEJEUNE	Gérard		
LEPROVOST	Christian		
MADAR	Roland		
MARTIN	Jean-Marie		
MERMET	Jean		

PERSONNES AYANT OBTENU LE DIPLOME
D'HABILITATION A DIRIGER DES RECHERCHES

BALESTRA	Francis
BALME	Louis
BECKER	Monique
BIGEON	Jean
BINDER	Zdeneck
BOE	Louis-Jean
CANUDAS DE WIT	Carlos
CHOLLET	Jean-Pierre
COEY	Jean-Pierre
CORNUEJOLS	Gerard
COURNIL	Michel
CRASTES DE PAULET	Michel
DALLERY	Yves
DESCOTES-GENON	Bernard
DUGARD	Luc
DURAND	Madeleine
FERRIEUX	Jean-Paul
FEUILLET	René
FREIN	Yannick
GAUTHIER	Jean-Paul
GHIBAUDO	Gérard
GUILLEMOT	Nadine
GUYOT	Alain
HAMAR	Sylviane
HAMAR	Roger
HORAUD	Patrice
JACQUET	Paul
LATOMBE	Claudine
LE HUY	Hoang
LE GORREC	Bernard
LOZANO-LEAL	Rogelio
MACOVSKI	Mihail
MAHEY	Philippe
METAIS	Olivier
MONMUSSON-PICQ	Georgette
MORY	Mathieu
MULLER	Jean
MULLER	Jean-Michel
NGUYEN TRONG	Bernadette
NIEZ	Jean-Jacques
PERRIER	Pascal
PLA	Fernand
RECHENMANN	François
ROGNON	Jean-Pierre
ROUGER	Jean
ROUX	Jean-Claude
TCHUENT	Maurice

PERSONNES AYANT OBTENU LE DIPLOME

DE DOCTEUR D'ETAT INPG

ABDEL-RAZEK	Adel
AKSAS	Haris
ALLA	Hassane
AMER	Ahmed
ANCELLE	Bernard
ANGENIEUX	Gilbert
ATMANI	Hamid
AYEDI	Hassine Feri
A.BADR	Osman
BACHIR	Aziz
BALANZAT	Emmanuel
BALTER	Roland
BARDEL	Robert
BARRAL	Gérard
BAUDON	Yves
BAUSSAND	Patrick
BEAUX	Jacques
BEGUINOT	Jean
BELLISSANT née FUNEZ	Marie-Claire
BELLON	Catherine
BEN RAIS	Abdejettah
BERGER-SABBATEL	Gilles
BERNACHE-ASSOLANT	Didier
BEROVAL	Abderrahmane
BERTHOD	Jacques
BILLARD	Dominique
BLANC épouse FOULETIER	Mireille
BOCHU	Bernard
BOJO	Gilles
BOKSENBAUM	Claude
BOLOPION	Alain
BONNARD	Bernard
BORRIONE	Dominique
BOUCHACOURT	Michel
BRINI	Jean
BRION	Bernard
CAIRE	Jean-Pierre
CAMEL	Denis
CAPERAN	Philippe
CAPLAIN	Michel
CAPOLINO	Gérard
CASPI	Paul
CHAN-TUNG	Nam
CHASSANDE	Jean-Pierre
CHATAIN	Dominique
CHEHIKIAN	Alain
CHIRAMELLA	Yves
CHILO	Jean
CHUPIN	Jean-Claude
COLONNA	Jean-François
COMITI	Jacques
CORDET	Christian
COUDURIER	Lucien

COUTAZ	Jean-Louis
CREUTIN	Jean-Dominique
DAO	Trongtich
DARONDEAU	Philippe
DAVID	Bertrand
DE LA SEN	Manuel
DELACHAUME	Jean-Claude
DENAT	André
DESCHIZEAUX née CHERUY	Marie-Noëlle
DIJON	Jean
DOREMUS	Pierre
DUPEUX	Michel
EL ADHAM	Karim
EL OMAR	Fovaz
EL-HENNA'WY	Adel
ETAY	Jacqueline
FABRE épouse MAXIMOVITCH	Suzanne
FAURE-BONTE épouse MARET	Mireille
FAYIER	Denis
FAYIER	Jean-Jacques
FELIACHI	Movloud
FERVAL	Haj Hassan
FLANDRIN	Patrick
FOREST	Bernard
FORESTIER	Michel
FOSTER	Panayolis
FRANC	Jean-Pierre
GADELLE	Patrice
GARDAN	Yvon
GENIN	Jacques
GERVASON	Georges
GILORMINI	Pierre
GINOUX	Jean-Louis
GOUMIRI	Louis
GROC	Bernard
GROSJEAN	André
GUEDON	Jean-Yves
GUERIN	Jean-Claude
GUESSOUS	Anas
GUIBOUD-RIBAUD	Serge
HALBWACHS	Nicolas
HAMMOURI	Hassan
HEDEIROS SILIYEIRA	Hamilton
HERAULT	Jeanny
HÜNER	Claude
HUECKEL	Tomasz
IGNAT	Michel
ILIADIS	Athanasios
JANIN	Gérard
JERRAYA	Ahmed Amine
JUTTEN	Christian
KAHIL	Hassan
KHUONGQUANG	Dong
KILLIS	Andreas
KONE	Ali
LABEAU	Michel
LACAZE	Alain
LACROIX	Jean-Claude
LANG	Jean-Claude
LATHUILLERE	Chantal

LATY	Pierre
LAUGIER	Christian
LE CADRE	Jean-Pierre
LE GARDEYR	René
LE NEST	Jean-François
LE THIESSE	Jean-Claude
LEMAIGNAN	Clement
LEMUET	Daniel
LEYEQUE	Jean-Luc
LONDICHE	Henry
L'HERITIER	Philippe
MAGNIN	Thierry
MAISON	François
MAMWI	Abdullah
MANTEL épouse SIEBERT	Elisabeth
MARCON	Guy
MARTINEZ	Francis
MARTIN-GARIN	Lionel
MASSE	Dominique
MAZER	Emmanuel
MERCKEL	Gérard
MEUNIER	Jean
MILI	Ali
MOALLA	Mohamed
MODE	Jean-Michel
MONLLOR	Christian
MONTELLA	Claude
MORET	Frédéric
MRAYATI	Mohammed
M'SAAD	Mohammed
M'SIRDI	Kouider Nace
NEPOMIASTCHY	Pierre
NGUYEN	Trong Khoi
NGUYEN-XUAN-DANG	Michel
ORANIER	Bernard
ORTEGA MARTINEZ	Roméo
PAIDASSI	Serge
PASSERONE	Alberto
PEGON	Pierre
PIJOLAT	Christophe
POGGI	Yves
POIGNET	Jean-Claude
PONS	Michel
POU	Tong Eck
RAFINEJAD	Paiviz
RAGAIE	Harie Fikri
RAHAL	Salah
RAMA SEABRA SANTOS	Fernando
RAYAINE	Denis
RAZBAN-HAGHIGHI	Tchanguiz
RAZZOUK	Micham
REGAZZONI	Gilles
RIQUET	Jean-Pierre
ROBACH	Chantal
ROBERT	Yves
ROGEZ	Jacques
ROHMER	Jean
ROUSSEL	Claude
SAAD	Abdallah
SAAD	Youcef

SABRY
SALON
SAUBAT épouse MARCUS
SCHMITT
SCHOELLKOPF
SCHOLL
SCHOLL
SCHOULER
SCHWARTZ
SEGUIN
SIWY
SKALLI
SKALLI HOUSSEYNI
SOUCHON
SUETRY
TALLAJ
TEDJAR
TEDJINI
TEYSSANDIER
THEVENODFOSSE
TMAR
TRIDLLIER
TUFFELIT
TZIRITAS
VALLIN
VELAZCO
VERDILLON
VERMANDE
VIKTOROVITCH
VITRANT
WEISS
YAZAMI

Mohamed Nabi
Marie-Christine
Bernadette
Jean-Hubert
Jean-Pierre
Michel
Pierre-Claude
Edmond
Jean-Luc
Jean
Jacques
Abdellatif
Abdelali
Alain
Jean
Nizar
Farouk
Smail
Francis
Pascale
Mohamed
Michel
Denis
Georges
Didier
Raoul
André
Alain
Pierre
Guy
François
Rachid



Président de l'Université :

M. NEMOZ Alain

ANNEE UNIVERSITAIRE 1990 - 1991

MEMBRES DU CORPS ENSEIGNANT DE SCIENCES ET DE GEOGRAPHIE

PROFESSEURS DE 1ERE CLASSE

ADIBA Michel
ANTOINE Pierre
ARVIEU Robert
AURIAULT Jean Louis
BARRA Jean René
BECKER Pierre
BEGUIN Claude
BELORISKY Elie
BENZAKEN Claude
BERARD Pierre
BERNARD Alain
BERTRANDIAS Françoise
BERTRANDIAS Jean Paul
BILLET Jean
BLANCHI Jean Pierre
BOEHLER Jean Paul
BOITET Christian
BORNAREL Jean
BRUANDET Jean François
CARLIER Georges
CASTAING Bernard
CHARDON Michel
CHIBON Pierre
COHEN ADDAD Jean Pierre
COLIN DE VERDIERE Yves
CYROT Michel
DEBELMAS Jacques
DEMAILLY Jean Pierre
GENEUVILLE Alain
DEPORTES Charles
DOLIQUE Jean Michel
DOUCE Roland
DUCROS Pierre
FINKE Gerde
GAUTRON René
GENIES Eugène
GERMAIN Jean Pierre
GIDON Maurice
GIGNOUX Claude
GILLARD Roland
GUITTON Jacques

Informatique
Géologie I.R.I.G.M.
Physique Nucléaire I.S.N.
Mécanique
Statistiques - Mathématiques Appliquées
Physique
Chimie Organique
Physique
Mathématiques Pures
Mathématiques Pures
Mathématiques Pures
Mathématiques Pures
Mathématiques Pures
Géographie
A.P.S.
Mécanique
Informatique et Mathématiques Appliquées

Physique
Biologie Végétale
Physique
Géographie
Biologie Animale
Physique
Mathématiques Pures
Physique du Solide
Géologie Générale
Mathématiques Pures
Physique
Chimie Minérale
Physique des Plasmas
Physiologie Végétale
Cristallographie
Informatique
Chimie
Chimie
Mécanique
Géologie
Sciences nucléaires
Mathématiques
Chimie

.. / ...

HERAULT Jeanny
 HICTER Pierre
 JANIN Bernard
 JOLY Jean René
 JOSELEAU Jean Paul
 KAHANE André
 KAHANE Josette
 KRAKOWIAK Sacha
 LAJZEROWICZ Jeanine
 LAJZEROWICZ Joseph
 LAURENT Pierre Jean
 LEBRETON Alain
 DE LEIRIS Joël
 LHOMME Jean
 LOISEAUX Jean Marie
 LONGEQUEUE Nicole
 LUNA Domingo
 MACHE Régis
 MASCLE Georges
 MAYNARD Roger
 NEMOZ Alain
 OMONT Alain
 PELMONT Jean
 PERRIER Guy
 PIERRE Jean Louis
 RENARD Michel
 RICHARD Jean Marc
 RIEDTMANN Christine
 RINAUDO Marguerite
 ROBERT Jean Bernard
 ROSSI André
 SAXOD Raymond
 SENDEL Philippe
 SERGERAERT Francis
 SOUCHIER Bernard
 STUTZ Pierre
 TRILLING Laurent
 VALLADE Marcel
 VAN CUTSEM Bernard
 VIALON Pierre
 VIDAL Micheal

Physique
 Chimie
 Géographie
 Mathématiques Pures
 Biochimie
 Physique
 Physique
 Mathématiques Appliquées
 Physique
 Physique
 Mathématiques Appliquées
 Mathématiques Appliquées
 Biologie
 Chimie
 Sciences Nucléaires I.S.N.
 Physique
 Mathématiques Pures
 Physiologie Végétale
 Géologie
 Physique du Solide
 Physique
 Astrophysique
 Biochimie
 Géophysique
 Chimie Organique
 Thermodynamique

Mathématiques
 Chimie C.E.R.M.A.V.

Biologie
 Biologie Animale
 Biologie Animale
 Mathématiques Pures
 Biologie
 Mécanique
 Mathématiques Appliquées
 Physique
 Mathématiques Appliquées
 Géologie

PROFESSEURS DE 2EME CLASSE

APPARU Marcel	Chimie
ARMAND Gilbert	Géographie
ARNAUD Hubert	Géologie
ARTRU Marie Christine	Physique
ATTANE Pierre	Mécanique
BARATE Robert	Sciences Nucléaires
BARET Paul	Chimie
BARGE Jean	Mathématiques
BARLET Roger	Chimie
BERTIN José	Mathématiques
BLOCK Marc	Biologie
BLUM Jacques	Mathématiques Appliquées
BOITET Christian	Mathématiques Appliquées
BORRIONE Dominique	Automatique informatique
BOULON Marc	Mécanique
BOUTRON Claude	Glaciologie
BOUVET Jean	Biologie
BROSSARD Jean	Mathématiques
BRUGAL Gérard	Biologie
CAMPILLO Michel	Géophysique
CAVILLE Jean Yves	Chimie
CERFF Rudiger	Biologie
CHIARAMELLA Yves	Mathématiques Appliquées
CHOLLET Jean Pierre	Mécanique
COLOMBEAU Jean François	Mathématiques (ENSL)
COTTET Georges-Henri	Modélisation, calcul scientifique, statis.
COURT Jean	Chimie
CUNIN Pierre Yves	Informatique
DAVID Jean	Géographie
DEROUARD Jacques	Physique
DHOUAILLY Danielle	Biologie
DUFRESNOY Alain	Mathématiques Pures
DUPUY Claude	Chimie
DURAND Mireille	Sciences Nucléaires
FONTECAVE Marc	Chimie
FOURNIER Jean Marc	Physique
GASPARD François	Physique
GIDON Maurice	Géologie
GIORNI Alain	Sciences Nucléaires
GONZALEZ SPRINBERG Gérardo	Mathématiques Pures
GOURC Jean Pierre	Mécanique
GUIGO Maryse	Géographie
GUMUCHIAN Hervé	Géographie
HACQUES Gérard	Mathématiques Appliquées
HAMMOU Abdelkader	Chimie
HERBIN Jacky	Géographie
HERINO Roland	Physique
HERZOG Michel	Biologie
JARDON Pierre	Chimie
JUTTEN Christian	Physique
KERCKHOVE Claude	Géologie
KOSAREW Siegmund	Math. fondamentales et appliquées
KLINGER Jurgen	Glaciologie
LAURENT Christine	Mathématiques
MANDARON Paul	Biologie
MARTINEZ Francis	Mathématiques Appliquées
MERCHEZ Fernand	Physique
MILAS Michel	Chimie
MOREL Alain	Géographie
MORIN Pierre	Physique
NGUYEN HUY Xuong	Informatique
OUDET Bruno	Mathématiques Appliquées

PAUTOU Guy
PECHER Arnaud
PELLETIER Guy
PERRIN Claude
PFISTER Claude
PIBOULE Michel
PORTESEIL Jean Louis
PUECH Laurent
RAYNAUD Hervé
REGNARD Jean René
ROBERT Claudine
ROBERT Danielle
ROBERT Gilles
SAJOT Gérard
SARROT REYNAULD Jean
SAYETAT Françoise
SERVE Denis
STOECKEL Frédéric
SCHOLL Pierre Claude
SUBRA Robert
TEMPERVILLE André
TISSUT Michel
TOURNIER Evelyne
VALLADE Marcel
VALLON Michel
VICAT Jean
VINCENS Maurice
VINCENT Gilbert
VIVIAN Robert
VOTTERO Philippe
WITOMSKI Patrick

Biologie
Géologie
Astrophysique
Sciences Nucléaires I.S.N.
Biologie
Géologie
Physique
Physique
Mathématiques Appliquées
Physique
Didactique des disciplines scientifiques
Chimie
Mathématiques Pures
Physique
Géologie
Physique
Chimie
Physique
Mathématiques Appliquées
Chimie
Mécanique
Biologie
Informatique et Mathématiques appliquées
Physique
Glaciologie
Physique
Chimie
Physique
Géographie
Chimie

Dédicace
&
Remerciements

à ma femme :
Isabelle.

REMERCIEMENTS

Je tiens ici à exprimer mes remerciements à :

- Monsieur Robert PERRET, Professeur à l'ENSIEG, qui m'a accueilli au sein du LEG, qui m'a fait l'honneur et le plaisir de présider le jury de cette thèse et qui a su m'apporter son aide en maintes circonstances.
- Monsieur Norbert GIAMBIASI, Professeur à l'EERIE de Nîmes, pour s'être intéressé à mon travail et avoir accepté d'être rapporteur de cette thèse.
- Monsieur Charles SOL, Professeur à l'ENSET de Cachan, pour l'intérêt qu'il a manifesté pour ce travail en acceptant d'être rapporteur de cette thèse.
- Monsieur Yves BAUDON, Maître de conférence à l'UJF de Grenoble, pour les précieuses relectures de ce rapport et nos discussions amicales sur notre Vendée natale.
- Monsieur Jean BIGEON, Chargé de recherche au CNRS, pour les nombreux conseils et encouragements qu'il m'a prodigués au cours de ce travail, qu'il trouve ici l'expression de ma profonde gratitude.
- Monsieur Gérard CHAMPENOIS, Maître de conférence à l'ENSPG, qui sans ménager ses efforts et sa disponibilité, s'est impliqué fortement dans le co-encadrement de cette thèse, qu'il soit assuré de ma sincère reconnaissance.

Je tiens à remercier aussi :

- Jean-Claude SABONNADIÈRE, directeur de LEG, pour ses encouragements et la confiance qu'il m'accorde.
- Messieurs Jean DELLA DORA et Gilles VILLARD pour m'avoir accueilli quelques mois au LMC et m'avoir permis généreusement de travailler sur "Alpe".
- Alain BOLOPION, maître incontesté de MC2, pour sa générosité, ses nombreux conseils et l'intérêt qu'il a porté à mon travail, qu'il trouve ici toute ma profonde reconnaissance

Remerciements

- Elisabeth RULLIERE, avec laquelle je partage le même bureau, qu'elle trouve ici l'expression de toute l'amitié que je lui porte.
- Marie-Thérèse LOUBINOUX, pour sa gentillesse et son dévouement.
- Daniel ROYE et Jean-Pierre ROGNON, les "tontons flingueurs" de la commande.
- James ROUDET, pour les conseils qu'il m'a apportés.
- Jacqueline DELAYE, Josiane EVEN, Sylvie PELLETIER et Sandrine, pour les différents services qu'elles m'ont rendus au cours de ces trois années de thèse.
- "LES FAOU", pour leur soutien.
- Bruno MALLET et toutes les personnes de la plateforme, pour l'ambiance chaleureuse et la bonne humeur qui y règnent.
- Toutes les personnes que j'ai pu côtoyer de près ou de loin au cours de ces trois années, dont les compagnons du "RU" et la bande de joyeux thésards du LEG. En particulier, les DEA et thésards du groupe IA et de l'équipe commande avec lesquels j'ai partagé de nombreux moments studieux, mais aussi des soirées et randonnées mémorables, dont les bivouacs dans Belledonne ! Je vous garde toute ma sympathie.

Enfin, je remercie mes parents pour m'avoir permis de rompre le cycle qui fait que l'on est agriculteur de père en fils, afin de réaliser des études. Qu'ils soient ici remerciés pour toute la confiance qu'ils me portent.

Sommaire

SOMMAIRE

GLOSSAIRE	p V
INTRODUCTION	p 1
CHAPITRE I : PROBLEMATIQUE DE LA CONCEPTION DES AMCC	p 3
A. Présentation générale du processus de conception	p 3
1) Les différentes étapes	p 3
2) Les contraintes	p 5
3) Les systèmes d'aide à la conception	p 6
B. Les logiciels de simulation d'AMCC	p 7
1) Les exigences d'un outil de simulation d'AMCC	p 7
2) Spécificités des outils de simulation existants	p 9
3) Limitations des outils de simulation existants	p 14
C. Système d'aide à la conception des AMCC	p 18
1) Objectifs du système expert	p 18
2) Caractéristiques du système expert	p 19
3) Structure d'un système expert de conception	p 20
CHAPITRE II : DEMARCHE PROPOSEE EN SIMULATION NUMERIQUE D'AMCC	p 23
A. Démarche proposée pour l'utilisateur	p 24
1) Le bloc : unité de base de la description	p 24
2) Méthodologie de description en blocs d'une AMCC	p 26
B. Traitement de la partie commande numérique	p 29
1) Aspects à simuler	p 29
2) Description des blocs TS _n	p 30
3) Résolution séquentielle des blocs	p 30
C. Traitement de la partie puissance et commande analogique	p 31
1) Aspects à simuler	p 31
2) Description des blocs	p 32
3) Résolution globale des équations	p 34
4) Gestion de l'enchaînement des configurations	p 42
D. Découplage entre boucles	p 42
1) Objectifs	p 42

II

2) Découplage par les blocs	p 42
3) Découplage par les constantes de temps	p 43
CHAPITRE III : PRESENTATION DE NOTRE OUTIL DE SIMULATION	p 45
A. Organisation de l'outil de simulation	p 45
1) Organisation d'une étude d'AMCC	p 45
2) Partie générateur	p 46
3) Partie moteur de résolution	p 47
4) Partie interface utilisateur	p 47
B. Utilisation de Macsyma	p 48
1) Le calcul formel	p 48
2) Intérêt du langage de calcul formel	p 48
3) Macsyma	p 50
C. Le générateur	p 50
1) Structure générale	p 50
2) Entrée des données	p 50
3) Réorganisation des équations	p 52
4) Génération automatique de code informatique	p 53
5) Tests sur les équations	p 53
6) Gestion de la conduction discontinue	p 54
7) Tests sur l'entrée des données (erreurs de description)	p 57
D. Ouvertures offertes par l'emploi du calcul formel	p 59
1) Ouverture à partir des équations réorganisées par notre générateur	p 59
2) Utilisation du calcul formel pour des calculs complexes	p 60
3) Bilan sur le calcul formel	p 61
E. Quelques chiffres sur notre générateur	p 61
CHAPITRE IV : APPLICATION ET RESULTATS	p 63
A. Machine à courant continu alimentée par un hacheur quatre quadrants commandé en modulation de largeur d'impulsion	p 63
1) Description du problème	p 64
2) Spécificités du problème	p 66
3) Résultats	p 68
B. Machine synchrone autopilotée alimentée en courant	p 71
1) Description du problème	p 72
2) Spécificités du problème	p 73
3) Résultats	p 75
C. Bilan sur notre approche en simulation	p 77

III

1) Interface utilisateur et portabilité	p 77
2) La nécessité de connaître un langage de programmation	p 78
3) Avantages	p 78
4) Bilan par rapport aux outils de simulation classiques d'AMCC	p 79
Conclusion	p 80

CHAPITRE V : MODELISATION DU PROCESSUS DE CONCEPTION DES AMCC

	p 82
A. Démarche de conception proposée	p 82
B. Mise en forme de la connaissance	p 84
1) Architecture du système expert	p 84
2) La connaissance factuelle	p 84
3) La connaissance déductive	p 88
4) La connaissance stratégique	p 92
C. Utilisation d'outils d'analyse	p 97

CHAPITRE V : MISE EN OEUVRE DU SYSTEME EXPERT

	p 99
A. Mise en oeuvre des données	p 99
1) Le formalisme des objets dans Smeci	p 99
2) Contraintes sur les objets	p 99
B. Mise en oeuvre des règles et des tâches	p 101
1) Le formalisme des règles dans Smeci	p 101
2) Mise en oeuvre des tâches	p 103
C. Mise en oeuvre de la stratégie de conception	p 105
1) Représentation de la stratégie	p 105
2) Simplification de l'arbre de raisonnement	p 105
3) Mise en oeuvre du raisonnement dans les étapes	p 107
4) Comparaison de contextes	p 110
5) Le suivi du raisonnement de conception	p 111
6) Travail effectivement réalisé	p 111
D. Une session avec le système expert	p 112
1) Elaboration du cahier des charges	p 112
2) Génération des différentes structures	p 112
3) Evaluation qualitatives des performances	p 114
4) Détails de structure et dimensionnement	p 116
5) Analyse des performances	p 116
6) Prototype et étude des solutions	p 117
E. Résultats	p 117

IV

F. Apports du système expert	p 119
CONCLUSION	p 121
ANNEXES :	
Annexe I : MC2	p 123
Annexe II : Macsyma	p 126
Annexe III : Entrée des données d'un problème avec notre générateur	p 129
Annexe IV : Code produit par notre générateur pour MC2	p 136
Annexe V : Modèle de machine synchrone avec amortisseurs	p 142
Annexe VI : Simulation de la commande numérique	p 146
Annexe VII : Découplage des boucles par des blocs	p 151
Annexe VIII : Modes d'exécution des règles	p 154
BIBLIOGRAPHIE	p 158

Glossaire

GLOSSAIRE

(L) : Indique une définition issue du Petit Larousse.

(R) : Indique une définition issue du Petit Robert.

AMCC : Association Machine - Convertisseur - Commande.

Approximatif : Se dit de ce qui est imprécis et incertain.

Causalité : L'effet ne peut pas avoir lieu avant la cause qui lui donne naissance.

Champs : Correspondent aux propriétés et caractéristiques des objets. Les champs caractérisent l'état des objets pendant l'exécution du système expert.

Classe : Concept générique qui permet de représenter un ensemble d'objets ayant la même structure.

Connaissance déductive : Elle représente le raisonnement humain dans le processus de perception, de prise de décision, de planification, de diagnostic, d'interprétation des données, de compréhension du langage et de conception [HAT 91].

Connaissance factuelle : Elle englobe l'ensemble des données physiques manipulées dans le système expert.

Connaissance stratégique : Elle représente la stratégie selon laquelle s'exécutent les règles issues de la connaissance déductive.

Contexte : Ensemble des objets qui existent à un instant donné du raisonnement, ainsi que la valeur de leurs champs.

Donnée (L) : Élément fondamental servant de base à un raisonnement.

Fait = Donnée.

Forme de réorganisation des équations :

Forme (1) : Forme classique des équations d'état (forme canonique),

Forme (2) : Forme triangularisée,

Forme (3) : Forme matricielle pour une résolution numérique de $\dot{X}_1$ et Y_1 .

Formule :

$A = [a_{ij}]_{n,m}$: Matrice de coefficients a_{ij} , à n lignes et m colonnes,

$V = [v_i]_n$: Vecteur de coefficients v_i , de dimension n ,

$\dot{X} = dX/dt$: Dérivée temporelle de X .

Générique (L) : Se dit d'un mot dont le sens englobe toute une catégorie d'êtres ou d'objets.

Heuristique (R) : Du grec heuriskien "trouver", qui sert à la découverte.

Imprécis : Qui manque de précision, concerne des termes tels que : fort, faible, petit, très grand...

Incertain : Qui est vague, concerne des termes tels que : plutôt, préférable, peut-être, vraisemblablement...

Inférence (L) : Opération intellectuelle par laquelle on passe d'une vérité à une autre vérité, jugée telle en raison de son lien avec la première. La déduction est une inférence.

Instanciable : Se dit d'une règle de production dont la partie "condition" est vérifiée.

Itératif (L) : Fait ou répété plusieurs fois.

MC2 : Logiciel de simulation d'AMCC développé au LEG (Machine - Convertisseur - Commande).

Moteur d'inférence : C'est le module du système expert qui permet d'enchaîner les règles suivant la stratégie qui a été fournie par le concepteur du système.

Objets : Correspond aux données manipulées par les règles issues de la connaissance déductive.

Règle de production : Règle du type "si condition alors action".

VII

Stratégie (L) : Art de coordonner des actions, de manoeuvrer habilement, pour atteindre un but.

Symbole : Chaîne de caractères sans "blanc"
ex : "ceci-est-un-symbole", "ceci n'est pas un symbole".

Introduction

INTRODUCTION

Au cours de ces dernières années, les progrès de l'électronique de puissance et de la commande numérique ont permis une évolution importante de la conception des associations machine électrique - convertisseur statique - commande (AMCC). Leur champ d'applications ne cesse de s'accroître et les facilités offertes maintenant par les logiciels de simulation de tels ensembles permettent des études de pointe comme les actionneurs pour la robotique, les machines outils, ou les nouveaux modes de traction électrique ...

Les cahiers des charges de ces applications sont définis en fonction d'exigences très strictes et variées qui sont souvent difficiles à concilier, comme par exemple des contraintes de coût en même temps que celles de performances. En fonction des critères définis pour une application donnée, plusieurs structures d'association machine - convertisseur - commande peuvent parfois convenir, la difficulté réside alors dans le choix de celle qui conduit à la solution optimale.

Devant la complexité de ces études d'optimisation, nous nous proposons d'apporter une aide au concepteur de tels ensembles en lui fournissant une méthodologie d'analyse lui permettant d'aboutir à une simulation de son problème. Pour prendre en charge les aspects itératifs et systématiques de la conception, nous lui proposons une approche par système expert.

Dans le chapitre I, après avoir analysé succinctement le processus de conception des AMCC, nous donnons les exigences de tels outils, puis nous en définissons les caractéristiques et les objectifs.

Ensuite, nous présentons quelques généralités sur les systèmes experts et la décomposition de la connaissance.

Dans le chapitre II, nous proposons une démarche de simulation numérique des AMCC, qui offre les avantages des outils existants, tout en soulevant leurs limites qui sont présentées au chapitre I. Cette méthodologie repose sur l'emploi d'une description modulaire des AMCC et d'une résolution globale de leurs équations.

Dans le chapitre III, nous présentons le générateur automatique d'équations et de programmes de simulation que nous avons élaboré à partir de cette démarche. Son originalité repose sur le fait qu'il ait été conçu à partir d'un langage de calcul formel.

Dans le chapitre IV, nous montrons une validation expérimentale de cet outil sur plusieurs applications traitées au sein du Laboratoire. Nous terminons ce chapitre par une présentation des principales performances que nous avons obtenues.

Disposant d'un outil de simulation, nous avons pu entreprendre l'élaboration d'un système expert d'aide à la conception des AMCC.

Celui-ci s'adresse au non spécialiste qui souhaite se familiariser avec le processus de conception. Notamment, il propose une aide à la caractérisation d'une application (élaboration du cahier des charges) ainsi qu'au choix des structures des associations pouvant être utilisées.

Dans le chapitre V, nous proposons une démarche de conception des AMCC dont nous avons cherché à systématiser certaines étapes à l'aide du système expert. Ensuite, nous présentons les différents concepts que nous utilisons pour modéliser la connaissance, en particulier l'emploi du formalisme orienté objets et de règles de production.

Nous avons mis ceux-ci en oeuvre à l'aide d'un générateur de système expert. Cette élaboration est présentée au début du chapitre VI. Ensuite nous montrons comment s'effectue une session de conception avec notre système expert. Puis, nous terminons ce chapitre par des essais de choix de structures d'AMCC avec le système, à partir de différents cahiers des charges d'application.

Chapitre I

Problématique de la conception des

associations machine - convertisseur - commande

CHAPITRE I

PROBLEMATIQUE DE LA CONCEPTION DES AMCC

Après une présentation rapide du processus de conception des AMCC (§ I.A.1) et des contraintes que celui-ci impose (§ I.A.2), nous montrons comment on peut aider (§ I.A.3) le concepteur d'AMCC en lui proposant un outil de simulation et un système d'aide à la conception.

A. PRESENTATION GENERALE DU PROCESSUS DE CONCEPTION

1) LES DIFFERENTES ETAPES

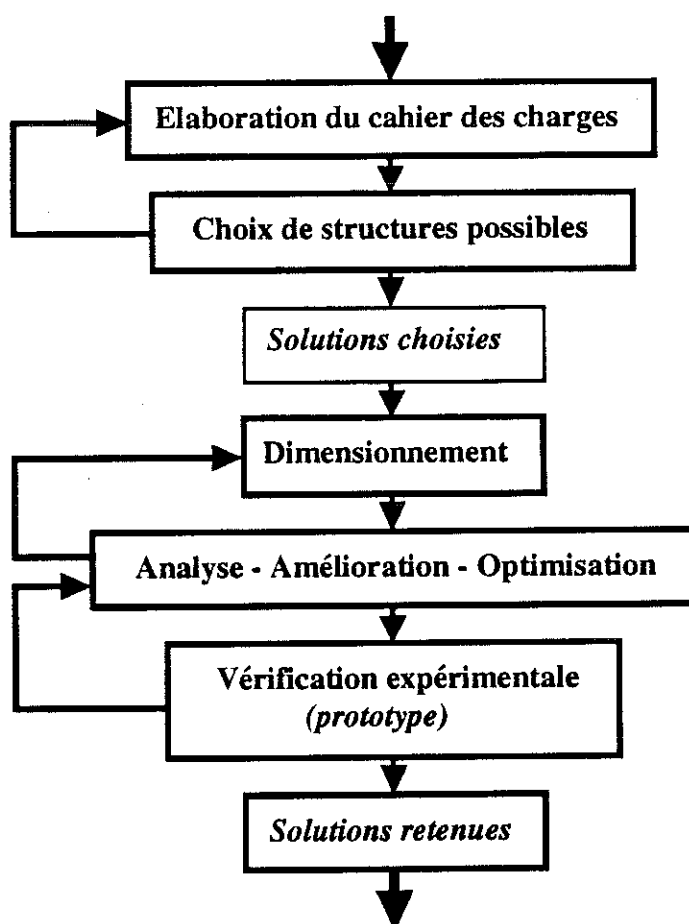


Figure I.1 : Exemple de démarche générale de conception d'AMCC

- Dans un premier temps, le client du concepteur d'AMCC élabore un **cahier des charges** de son application (robotique, traction, levage, ...), à partir :

- de ses spécificités (contrôle de couple, suivi de trajectoire, contrôle de position, ...),
 - de ses contraintes (coût, rendement, couple, puissance, environnement ...).
- Le concepteur utilise alors ces données pour établir un **choix grossier de la structure** de l'AMCC à concevoir pour éventuellement lui affecter certaines caractéristiques (type de machine, ...). Dans ce but, il effectue des calculs simples et il fait appel à un savoir faire (connaissance de nature heuristique). Il emploie pour cela des **règles d'expertises** qui sont parfois incertaines, imprécises et qualitatives, et des **modèles simplifiés**. Ainsi, il est obligé de faire des hypothèses et des choix qui peuvent être **remis en cause** dans une phase ultérieure.
- Ensuite, le concepteur **définit de façon précise et dimensionne** les différents composants de son AMCC en s'aidant d'outils d'analyse numérique (CAO, outils de calcul, outils de simulation). Dans cette étape, il doit prendre en compte les interactions qui peuvent exister entre les différents composants de l'association. Par exemple, les harmoniques, dus aux convertisseurs statiques, engendrent des pertes dans la machine qui peuvent contraindre le concepteur à prévoir une puissance supérieure à celle souhaitée pour cette machine.
- Remarque :** en général, le concepteur cherche à utiliser des composants issus des catalogues des fabricants.
- Pour assurer une bonne cohérence entre ces composants, le concepteur a souvent recours à une **analyse par la simulation numérique**. Cette dernière lui permet d'affiner la commande, de vérifier les performances demandées avant de passer à l'élaboration d'un prototype. Lorsque plusieurs solutions sont analysées en parallèle, cette phase permet de sélectionner les meilleures solutions qui devront être testées expérimentalement.
- Le concepteur combine ensuite l'emploi du **prototype** et de la simulation numérique, pour améliorer la future association à concevoir.
- Enfin, le concepteur effectue une **validation expérimentale** qui est l'ultime étape permettant d'apprécier précisément les performances des AMCC conçues.

Remarques : Ces phases de validation et d'analyse peuvent conduire le concepteur à revoir partiellement ou totalement ses choix.

2) LES CONTRAINTES

Le processus de conception est caractérisé par différentes contraintes et aspects que nous présentons ici :

a) Recherche du moindre coût

Le concepteur doit souvent tenter d'obtenir l'AMCC la moins chère qui répond aux contraintes du cahier des charges de son client. Cependant, pour des applications de pointe, les critères de rendement, d'encombrement ou de performances de fonctionnement peuvent être plus importants.

b) Certaines phases du processus sont itératives

Dans le processus de conception, le concepteur doit faire des choix, les tester, puis les modifier s'il n'obtient pas de résultats satisfaisants.

Ces choix se font dans différentes étapes du processus de conception. Par exemple, si à partir d'un **cahier des charges incomplet**, le concepteur fournit une AMCC avec une machine à courant continu (donc avec un collecteur) et qu'ensuite le client lui annonce que le milieu où doit fonctionner cette association est explosif, alors le concepteur devra obligatoirement revoir ses choix. Dans ce cas particulier, il devra proposer une nouvelle AMCC avec une machine sans collecteur.

Des retours en arrière se font aussi à l'intérieur d'une même étape. Par exemple, dans la phase de réglage et d'affinage des régulations (étape de dimensionnement), le concepteur teste sa commande et ses gains de régulations, puis il les modifie jusqu'à obtenir les performances qu'il désire.

c) Utilisation de règles heuristiques

Dans les différentes étapes du processus de conception, le concepteur utilise des règles issues de l'heuristique. Elles traduisent souvent, de façon simple, synthétique et qualitative le bon sens de l'expert, mais aussi des connaissances issues de l'expérience et de travaux d'analyse généralement longs et complexes.

Remarque : Cette expertise est souvent caractérisée par des termes qualitatifs (faible, fort, ...) et des nuances du type "plutôt", "préférable" ...

d) Evolution de l'expertise

L'expertise est en évolution constante, ce qui peut conduire à des révisions parfois fondamentales sur les choix d'une structure d'AMCC par rapport à une autre, pour une application donnée. Par exemple, en raison de l'évolution des coûts des composants et des progrès techniques et technologiques, certaines solutions, qui étaient rejetées à l'état de prototype pour des raisons économiques, peuvent devenir envisageables. Ainsi, en traction ferroviaire, la machine à courant continu est maintenant délaissée au profit des machines synchrones et asynchrones autopilotées.

e) Utilisation d'outils d'analyse

En raison de la complexité des AMCC, le concepteur à recours à des outils d'analyse performants pour les étapes de dimensionnement et d'analyse. En particulier, pour tester différentes commandes de machine et évaluer les performances de plusieurs AMCC répondant aux contraintes d'un même cahier des charges, la simulation s'avère un moyen rapide et efficace pour choisir les meilleures solutions.

3) LES SYSTEMES D'AIDE A LA CONCEPTION

Les systèmes d'aide à la conception à bases de connaissances (systèmes experts) sont de plus en plus employés en Génie Electrique. Ils permettent :

- de confronter facilement plusieurs expertises,
- de concevoir en parallèle plusieurs composants à partir d'un même cahier des charges (ce qui est difficile pour le concepteur qui conduit généralement une seule solution à la fois),
- de gérer automatiquement les aspects itératifs de la conception, qui peuvent être systématisés par l'emploi de modèles et de règles.

Ainsi, des systèmes experts ont été réalisés pour :

- la conception des convertisseurs statiques [AHR91] [HSI92] [CHH92], des machines asynchrones [TRI91], des contacteurs électriques [BRA89] [GEN91],
- l'amélioration de commande dans un cas précis d'AMCC [BEZ92],
- Le choix d'une AMCC dans un catalogue à partir d'un cahier des charges et d'une machine spécifique [CHE92], [PRA91].

A notre connaissance, rien n'a encore été fait sur les **systèmes d'aide à la conception d'AMCC de type système expert**. Dans le cadre de cette thèse, nous proposons de commencer une étude dans cet axe de recherche.

Cependant, avant de chercher à systématiser le processus de conception des AMCC (même partiellement), nous avons dû regarder si nous disposions des conditions minimales pour que ce processus soit effectué par le concepteur; en particulier : les outils actuels de simulation d'AMCC sont-ils satisfaisants ? Ainsi nous avons constaté que les outils classiques, qui sont performants pour de nombreuses applications, perdent leurs avantages pour d'autres.

B. LES LOGICIELS DE SIMULATION D'AMCC

Les outils de simulation d'AMCC permettent principalement d'étudier leur commande sur un modèle macroscopique représentatif du banc d'essais. Ils permettent aussi de faire des études sans risquer de destruction de composants et avec plus de flexibilité et de rapidité que l'expérience cependant indispensable pour valider ces études.

Avant de présenter les outils de simulation classiques, nous définissons nos exigences vis à vis de ces outils.

1) LES EXIGENCES D'UN OUTIL DE SIMULATION D'AMCC

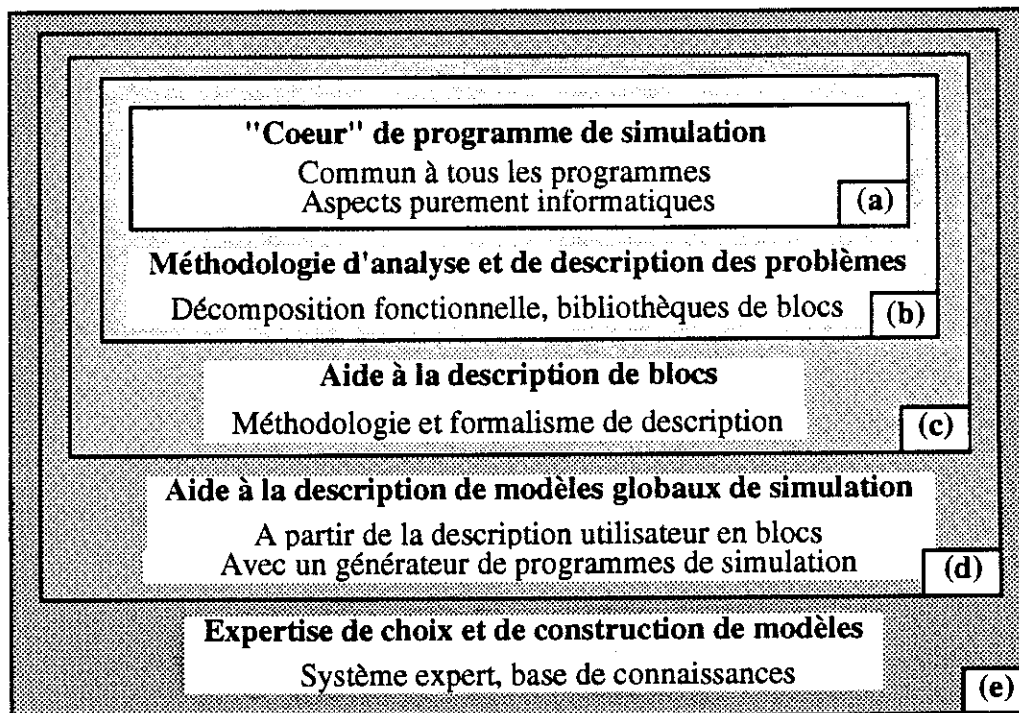


Figure I.2 : Niveaux de service d'un outil de simulation d'AMCC

Afin de limiter le temps de développement et de faciliter la création d'une nouvelle application de simulation, un outil de simulation d'AMCC doit offrir plusieurs services (figure I.2).

a) Un "cœur" d'outil de simulation

Il doit fournir à l'ingénieur, ou chercheur, un environnement le libérant des contraintes informatiques que l'on rencontre dans la plupart des programmes de simulation :

- une gestion de données (entrée et modification interactives),
- une gestion et une visualisation graphique des résultats de simulation en vue de leur exploitation,
- des méthodes de résolution implicites d'équations différentielles (cf § I.B.2.b).

Ceci correspond au cœur de tout programme de simulation.

b) Une méthodologie de description des AMCC

Il doit proposer une méthodologie permettant à son utilisateur d'avoir une analyse structurée de son problème. En particulier, il doit proposer une description en entités de base (blocs) des AMCC. Ceci permet :

- de faire un développement rapide de nouvelles applications par l'utilisation de blocs existant en bibliothèque,
- d'offrir la possibilité de modifier aisément une application de simulation, et de bénéficier de différents modèles existants pour un même sous-ensemble et donc de pouvoir comparer ceux-ci.

c) Une aide à la description des blocs

Pour que tout bloc puisse être facilement conçu, mais aussi modifié et compris au niveau de sa structure par tout autre utilisateur de l'outil (concepteur de blocs), il doit fournir un formalisme de description standardisé.

Tous les blocs créés sont stockés en bibliothèque.

d) Une aide à la construction de programmes de simulation

Un outil performant doit fournir une aide à la construction de modèles globaux de simulation.

Pour cela, il doit vérifier la cohérence des structures décrites, mais aussi aider l'utilisateur à construire des programmes de simulation rapidement, à partir de l'analyse et de la description en blocs qui aura été réalisée préalablement. Ceci correspond à un générateur de programmes.

e) Une expertise de simulation

Un dernier service serait qu'il fournisse une expertise de simulation permettant d'aider l'utilisateur à faire le bon choix de modèle pour chaque bloc de l'AMCC étudiée.

Cette expertise serait issue d'une part des applications déjà traitées et d'autre part de l'expérience des personnes utilisant l'outil de simulation. Il faut que de nombreuses applications de simulation aient déjà été traitées pour que l'on puisse envisager de structurer une telle expertise.

Remarque : Un outil de simulation d'AMCC doit répondre à des exigences issues du génie logiciel. En particulier, pour assurer son développement et pour faciliter la maintenance de l'outil, il faut veiller, lors de sa conception, à l'élaborer de façon modulaire, en discernant bien ses fonctionnalités (si possible : une par module).

2) SPECIFICITES DES OUTILS DE SIMULATION EXISTANTS

Les logiciels de simulation numérique d'AMCC les plus connus en France sont SOSIE [FAU89], GASPE [DOE88], MC2 [BOL90], SIMNON [BOS89] ... Ils sont caractérisés par les spécificités suivantes :

a) Approche modulaire

Ils présentent généralement une approche modulaire [DOE88], [SIM90], [BOW90], [BOL90], [BUR89], c'est-à-dire que leurs utilisateurs décomposent les AMCC en blocs fonctionnels représentant chacun une entité : machine, convertisseur statique, régulateur, ...

Ces blocs sont caractérisés par (figure I.3)

- des entrées,
- des paramètres (résistance, inductance, gains de régulateur ...),
- des sorties,
- des équations permettant de fournir les sorties à partir des paramètres et des entrées,
- des tests correspondant à des tops d'horloge, des tests de commutation des interrupteurs

des convertisseurs ...

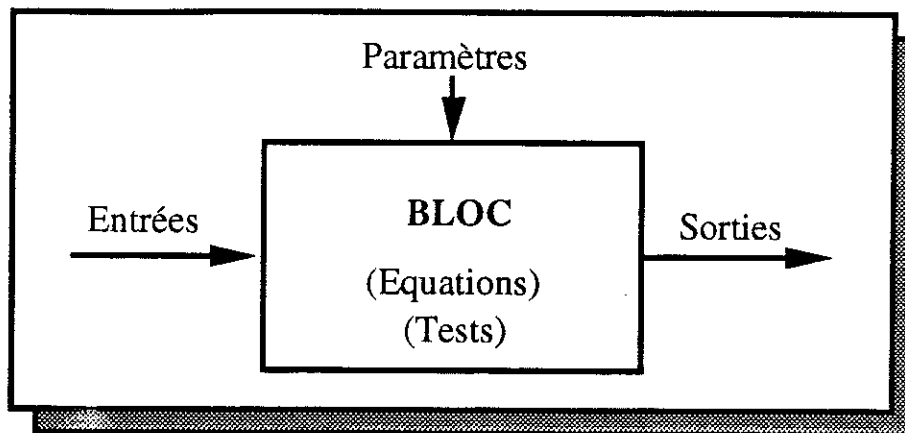


Figure I.3 : Aspect général d'un bloc

b) Méthode implicite de résolution différentielle

Les modèles utilisés pour les AMCC complètes sont souvent non linéaires (ex : modèles de machines électriques) et peuvent présenter des discontinuités (ex : modèles à topologie variable des convertisseurs statiques).

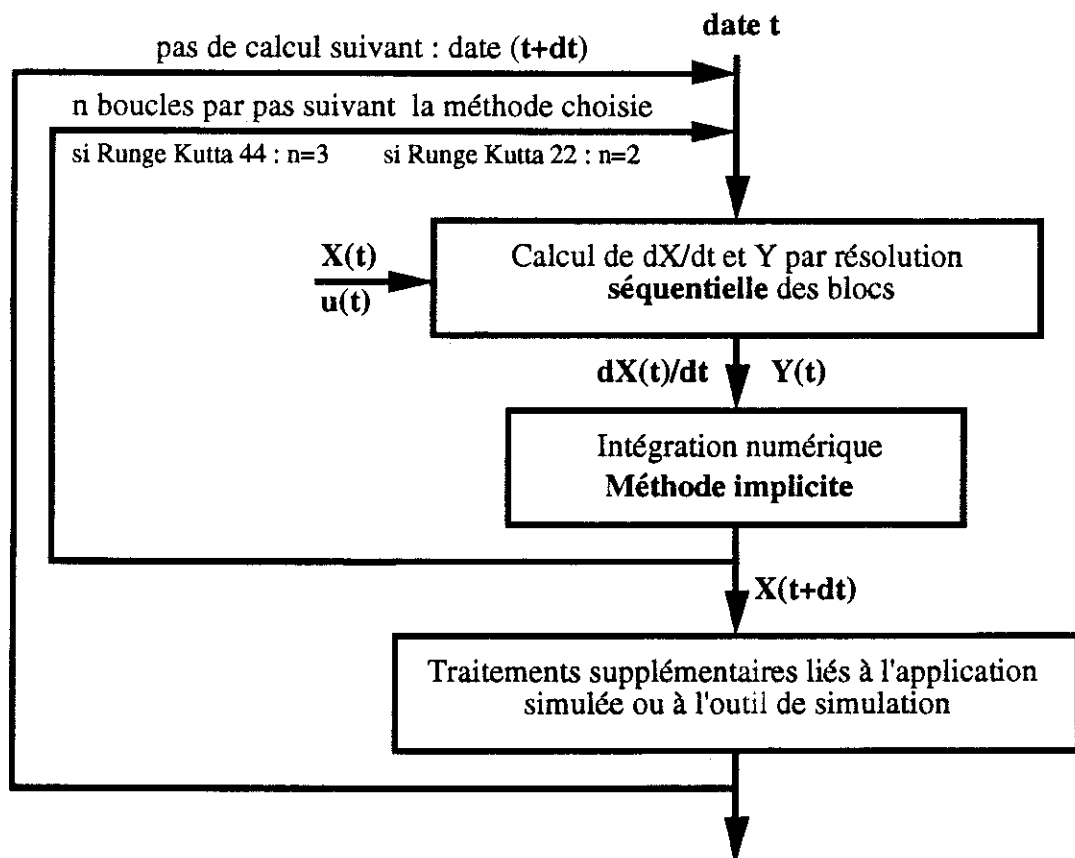


Figure I.4 : Résolution dynamique

Pour la simulation numérique de tels ensembles, on amené à utiliser des méthodes de résolution implicites d'équations différentielles [BOS90], [BUL88] : Euler (tangente), Adams, Bashfort, Runge Kutta (22,44). Elles permettent d'intégrer l'équation d'état de l'AMCC étudiée, qui s'écrit :

$$f(\dot{X}(t), X(t), Y(t), u(t), t) = 0 :$$

où : $X(t)$: vecteur d'état à l'instant t ,
 $\dot{X}(t) = dX(t)/dt$: dérivée temporelle de $X(t)$ à l'instant t ,
 $Y(t)$: vecteur de sortie à l'instant t ,
 $u(t)$: vecteur d'entrée (excitation) à l'instant t ,
 t : temps courant.

Le calcul de $dX(t)/dt$ et $Y(t)$ se fait généralement de façon séquentielle (figure I.4), par résolution successive des blocs [DOE88], [SIM90], [BOW90], [BOL90] suivant les connexions de leurs entrées / sorties. Par contre, l'obtention de $X(t)$ se fait de façon globale, par le module de résolution implicite de l'outil de simulation.

c) Gestion des événements importants

Pour améliorer la précision des calculs, il est important de détecter et de prendre correctement en compte les dates importantes qui correspondent soit à des changements de configuration de la partie puissance de l'association (tests de commutation d'interrupteurs), soit à des tops d'horloge (échantillonnage de la commande)...

Cette prise en compte se fait :

- par une méthode de prédiction qui utilise des pas variables (GASPE, MC2),
- par des méthodes de correction : si un événement important vient d'être franchi, un retour en arrière permet de recalculer la simulation (le pas de calcul) sur l'instant correspondant (SOSIE).

Ces dernières sont plus lourdes à mettre en oeuvre [NOR92].

d) Modèles à topologie variable des convertisseurs statiques

Des modèles à topologie variable [IUN] [DOE90] sont utilisés pour obtenir des temps de simulation les plus courts possibles et éviter les problèmes liés aux disparités des constantes de temps (cas des modèles à topologie fixe [ENJ89]). On obtient ainsi des modèles macroscopiques des blocs convertisseurs statiques qui représentent au mieux leur

comportement réel. Ces modèles sont constitués de différents jeux d'équations (un par configuration du convertisseur) et de tests permettant de gérer l'état de conduction des interrupteurs (donc l'enchaînement dynamique des configurations).

Le fonctionnement des convertisseurs statiques utilisés dans les AMCC étant souvent parfaitement connu, ces modèles sont entièrement définis avant la simulation, sous forme de blocs stockés en bibliothèque. Seules les configurations intervenant réellement dans le fonctionnement des AMCC sont modélisées.

Notons que ces interrupteurs peuvent être représentés par différents schémas selon le choix du modèle (figure I.5).

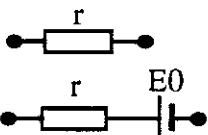
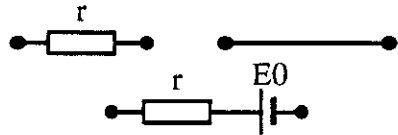
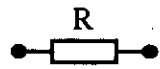
	Modèle avec topologie fixe	Modèle avec topologie variable
Etat passant	 r faible	 r
Etat bloqué	 R élevée supérieure à $10^5 r$	

Figure I.5 : Exemples de modèles d'interrupteurs

Les règles, permettant de gérer l'état des interrupteurs, correspondent à des tests sur leur commande ainsi qu'à des tests de franchissement de seuil pour les tensions et les courants. Cet enchaînement peut être formulé à l'aide de réseaux de Pétri [HAU] [SAH91]. A titre d'exemple, la figure I.6 présente le modèle à topologie variable du hacheur série, et rappelle sa représentation en topologie fixe.

Dans l'outil de simulation, pour chaque bloc, les tests sont écrits dans un programme qui affecte un indicateur de configuration en fonction de l'état du convertisseur. A tout instant, au cours de la simulation, cet indicateur permet d'assurer l'aiguillage vers le bon jeu d'équations (figure I.7).

Remarque : Une méthodologie de description de ces modèles a été développée et mise en oeuvre au sein du LEG [NOR92].

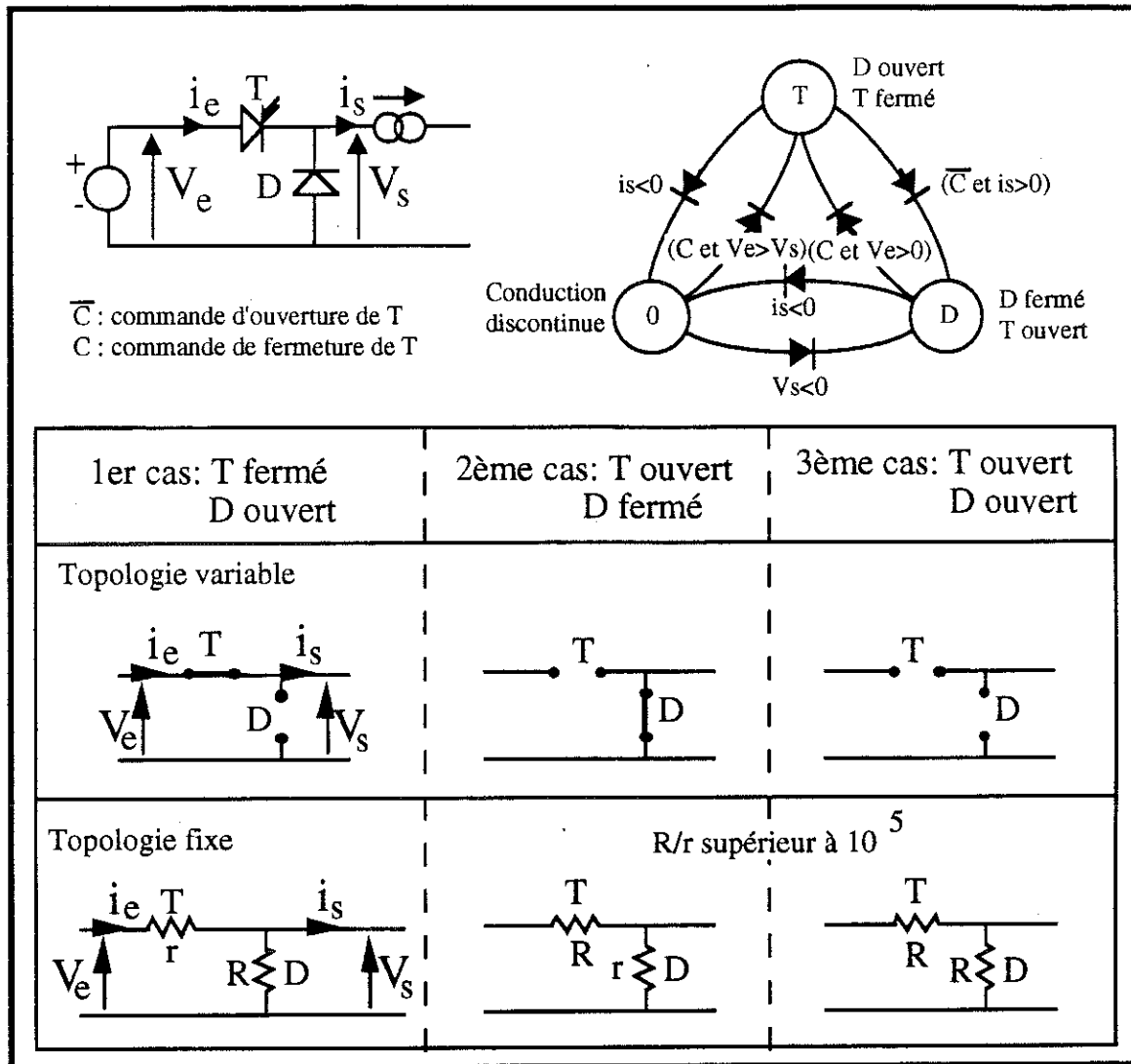


Figure I.6 : Différences entre topologie fixe et topologie variable : exemple du hacheur série

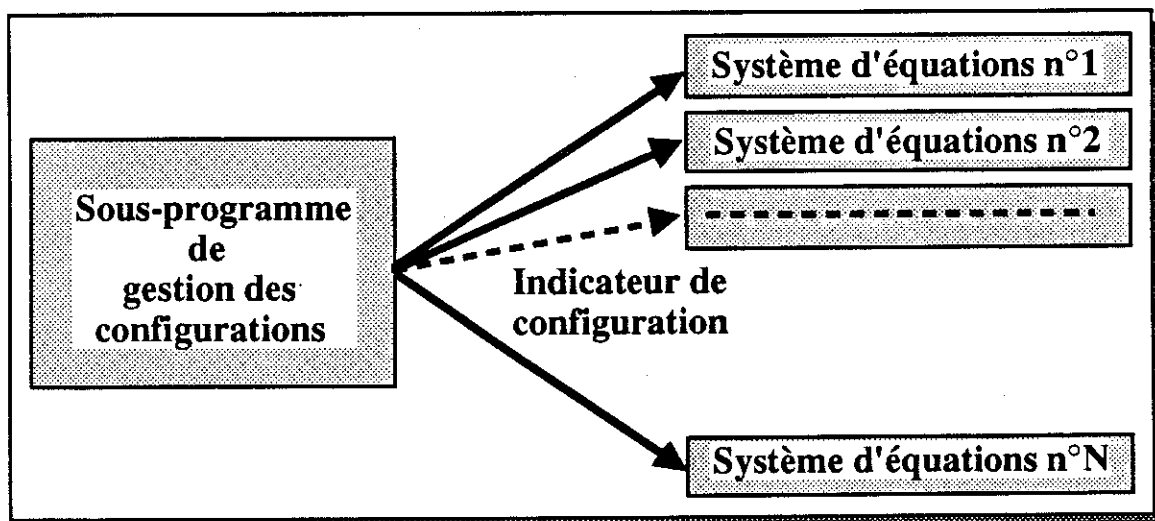


Figure I.7 : Bloc convertisseur statique : modèle à topologie variable

3) LIMITATIONS DES OUTILS DE SIMULATION EXISTANTS

Généralement, dans les outils de simulation classique, les blocs sont résolus séquentiellement à chaque pas de calcul [DOE88], [FAU89], [BOL88]. Nous allons montrer autour d'exemples simples que cette démarche séquentielle peut entraîner des problèmes de résolution et limiter fortement le niveau de décomposition en blocs des parties puissance et commande analogique des AMCC [GER90].

Remarque : Sur des exemples complexes tels que celui de la machine synchrone autopilotée (cf chapitre IV.B), l'utilisateur, avec des outils classiques, est obligé de faire un gros bloc pour la puissance de cette AMCC, donc il doit effectuer un travail considérable pour obtenir sa simulation [PAZ92]. Nous souhaitons soulever ces contraintes dans cette thèse.

a) Problème de la résolution modulaire séquentielle

Pour les parties puissance et commande analogique des AMCC, une résolution séquentielle et séparée des équations des différents blocs, peut conduire à un résultat faux. Nous allons le montrer autour de l'exemple simple de la figure I.8.

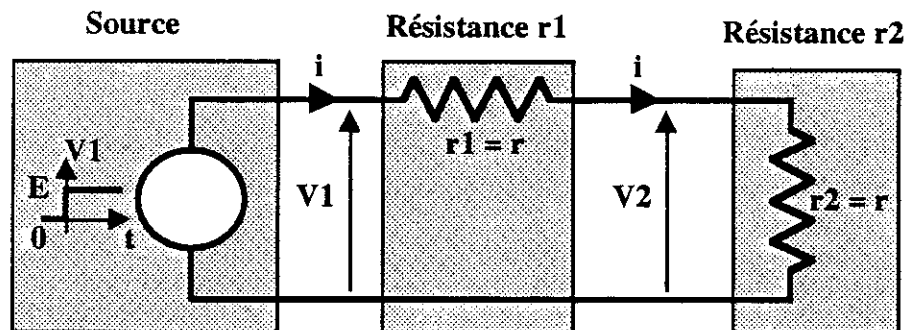


Figure I.8 : Exemple simple pour l'analyse de la résolution séquentielle

Ce système se compose de trois blocs :

- la source : $V_1 = E$,
- la résistance r_1 : $r_1 \cdot i = V_1 - V_2$,
- la résistance r_2 : $r_2 \cdot i = V_2$.

Si on résout séquentiellement ces trois blocs, on obtient, par exemples, plusieurs combinaisons possibles pour la résolution :

Ordre d'exécution du calcul à chaque pas Δt	$V_1 = \begin{matrix} 0 \text{ pour } t \leq 0 \\ E \text{ pour } t > 0 \end{matrix}$	$V_1 = \begin{matrix} 0 \text{ pour } t \leq 0 \\ E \text{ pour } t > 0 \end{matrix}$
	puis :	puis :
	$i = (V_1 - V_2) / r_1$	ou $V_2 = V_1 - r_1 \cdot i$
	puis :	puis :
	$V_e = r_2 \cdot i$	$i = V_2 / r_2$

où t représente le temps courant. Pour simplifier la démonstration, prenons $r_1 = r_2 = r$ et supposons le pas de calcul Δt constant.

Une résolution séquentielle donne :

à $t = 0$	$V_1(0) = 0$	$V_1(0) = 0$
	$i(0) = 0$	ou $V_2(0) = 0$
	$V_2(0) = 0$	$i(0) = 0$

au $(2n-1)^{\text{ième}}$ pas de calcul, c'est-à-dire à $t = (2n-1) \cdot \Delta t$

$V_1((2n-1) \cdot \Delta t) = E$	$V_1((2n-1) \cdot \Delta t) = E$
puis :	puis :
$i((2n-1) \cdot \Delta t) =$ $[V_1((2n-1) \cdot \Delta t) - V_2((2n-2) \cdot \Delta t)] / r = E/r$	$V_2((2n-1) \cdot \Delta t) =$ $V_1((2n-1) \cdot \Delta t) - r \cdot i((2n-2) \cdot \Delta t) = E$
puis :	puis :
$V_2((2n-1) \cdot \Delta t) = r \cdot i((2n-1) \cdot \Delta t) = E$	$i((2n-1) \cdot \Delta t) = V_2((2n-1) \cdot \Delta t) / r = E/r$

au $(2n)^{\text{ième}}$ pas de calcul, c'est-à-dire à $t = 2n \cdot \Delta t$

$V_1(2n \cdot \Delta t) = E$	$V_1(2n \cdot \Delta t) = E$
puis :	puis :
$i(2n \cdot \Delta t) = [V_1(2n \cdot \Delta t) - V_2((2n-1) \cdot \Delta t)] / r = 0$	$V_2(2n \cdot \Delta t) = V_1(2n \cdot \Delta t) - r \cdot i((2n-1) \cdot \Delta t) = 0$
puis :	puis :
$V_2(2n \cdot \Delta t) = r \cdot i(2n \cdot \Delta t) = 0$	$i(2n \cdot \Delta t) = V_2(2n \cdot \Delta t) / r = 0$

Bien que cette démarche séquentielle semble juste, elle conduit à un résultat global faux (figure I.9). Un résultat correct est obtenu en résolvant toutes ces équations ensemble, ce qui donne (figure I.10) :

$$V_1 = E$$

$$V_2 = E/2$$

$$i = E/(2r)$$

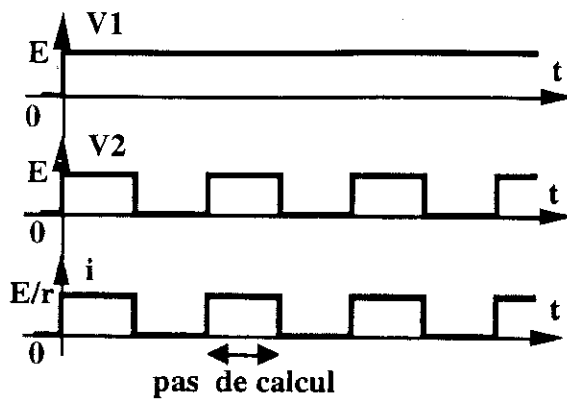


Figure I.9 : Résultat avec une résolution séquentielle

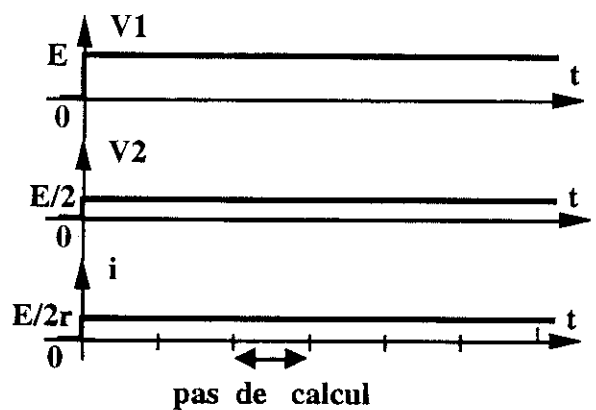


Figure I.10 : Résultat avec une résolution globale

Cette résolution globale traduit une réalité physique, car les grandeurs E , V_1 , V_2 et i évoluent en même temps.

b) Problème de causalité lié à l'approche modulaire

Lorsque l'on considère une AMCC, les aspects modulaire et séquentiel de la résolution induisent des problèmes de causalité (l'effet ne peut pas avoir lieu avant sa cause) pour le calcul des variables d'état [FAU91] [FAU92].

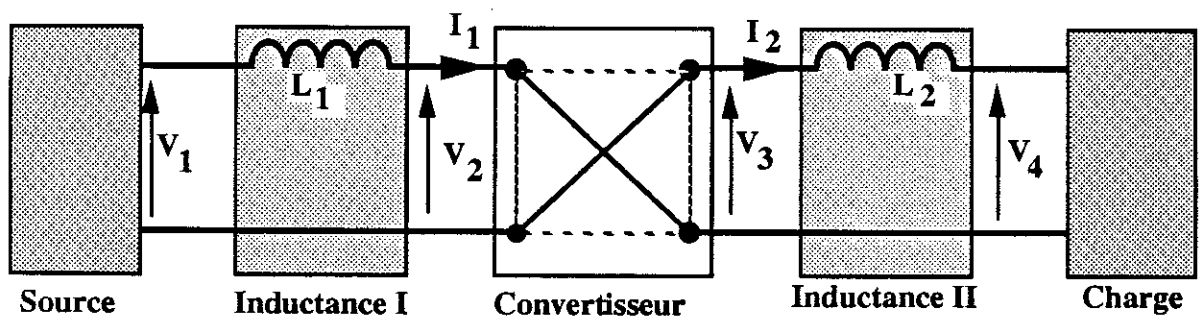


Figure I.11 : Exemple de boucle de puissance

Pour développer cet aspect, considérons l'exemple simple de la figure I.11. Ce système se compose de cinq blocs :

- la source V_1 fonction du temps courant,
- l'inductance I dont l'équation est $L_1 \cdot di_1/dt = V_1 - V_2$,
- le convertisseur modélisé en topologie variable (voir figure I.12) et dont les équations s'écrivent :

Configuration n°1	Configuration n°2	Configuration n°3	Configuration n°4
$i_1 = -i_2$ $V_2 = -V_3$	$i_1 = i_2$ $V_2 = V_3$	$V_2 = 0$ $V_3 = 0$	$i_1 = 0$ $i_2 = 0$

- l'inductance II dont l'équation est $L_2 \cdot di_2/dt = V_3 - V_4$,
- la charge dont l'équation sera $V_4 = f(i_2)$.

Les variables i_1 et i_2 représentent la même grandeur : $i = i_1 = -i_2$ dans la configuration n°1 et $i = i_1 = i_2$ dans la configuration n°2. Cependant, leurs dérivées sont calculées dans deux blocs différents : l'inductance I et l'inductance II. Ce problème de causalité ne peut être facilement résolu qu'en rassemblant et réorganisant l'ensemble des équations des divers blocs, en un seul et unique système d'équations pour chaque configuration de l'application étudiée.

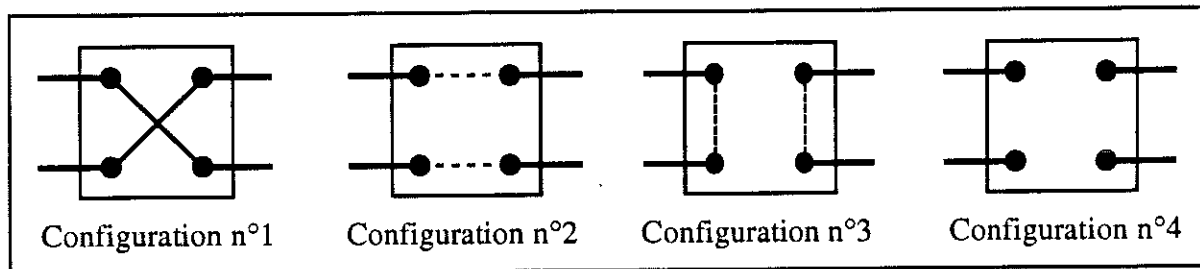


Figure I.12 : Configurations du convertisseur

Dans cet exemple, nous obtenons :

Configuration n°1	Configuration n°2	Configuration n°3	Configuration n°4
$\frac{di_1}{dt} = \frac{V_1 + V_3}{L_1}$ $V_4 = f(-i_1)$ $V_3 = V_4 - L_2 \frac{di_1}{dt}$ $V_2 = -V_3$ $i_2 = -i_1$	$\frac{di_1}{dt} = \frac{V_1 - V_3}{L_1}$ $V_4 = f(i_1)$ $V_3 = V_4 + L_2 \frac{di_1}{dt}$ $V_2 = V_3$ $i_2 = i_1$	$\frac{di_1}{dt} = \frac{V_1}{L_1}$ $V_4 = f(i_2)$ $\frac{di_2}{dt} = \frac{-V_4}{L_2}$ $V_2 = 0$ $V_3 = 0$	$V_1 = V_2$ $V_4 = f(i_2=0)$ $V_3 = V_4$ $i_2 = 0$ $i_1 = 0$

On constate que pour certaines configurations, cette réorganisation permet de réduire la taille des systèmes d'état (le nombre des variables différenciées).

A la suite de cette globalisation, une réorganisation finale doit être réalisée pour chaque configuration, afin que les systèmes d'équations soient directement utilisables par un module d'intégration différentielle de type pas à pas (ex : Runge Kutta).

Configuration n°1	Configuration n°2	Configuration n°3	Configuration n°4
$V_4 = f(-i_1)$ $\frac{di_1}{dt} = \frac{V_4 + V_1}{L_2 + L_1}$ $V_3 = \frac{L_1 V_4 - L_2 V_1}{L_2 + L_1}$ $V_2 = -V_3$ $i_2 = -i_1$	$V_4 = f(i_1)$ $\frac{di_1}{dt} = \frac{V_1 - V_4}{L_2 + L_1}$ $V_3 = \frac{L_1 V_4 + L_2 V_1}{L_2 + L_1}$ $V_2 = V_3$ $i_2 = i_1$	$V_4 = f(i_2)$ $V_2 = 0$ $V_3 = 0$ $\frac{di_1}{dt} = \frac{V_1}{L_1}$ $\frac{di_2}{dt} = -\frac{V_4}{L_2}$	$V_4 = f(i_2=0)$ $V_3 = V_4$ $V_2 = V_1$ $i_2 = 0$ $i_1 = 0$

Dans cette étude, nous proposons une méthodologie de simulation permettant d'éviter les problèmes que nous venons d'exposer tout en conservant les avantages des outils classiques (chapitre II). Cette démarche nous a permis de mettre au point un générateur automatique d'équations globales et de programmes de simulation qui sera présenté plus loin (chapitre III). L'outil obtenu ayant été validé sur plusieurs applications traités expérimentalement au laboratoire (chapitre IV), nous avons alors pu entreprendre une étude sur les systèmes experts dans le but de faciliter la conception des AMCC.

C. SYSTEME D'AIDE A LA CONCEPTION DES AMCC

Pour élaborer un système expert d'aide à la conception, nous nous sommes fixés les objectifs généraux suivants.

1) OBJECTIFS DU SYSTEME EXPERT

Le système expert doit offrir les services suivants.

- Il doit inclure une **base de connaissances**, laquelle doit pouvoir être enrichie et actualisée par tout expert de la conception des AMCC.
- Il doit proposer une **stratégie de conception**, qui reproduit au mieux la démarche d'un expert, en introduisant tous les avantages offerts par une approche informatique, en particulier en systématisant (automatisant) certaines étapes de ce processus de conception.
- Il doit permettre au **non spécialiste** de se familiariser avec les problèmes de conception d'AMCC. Notamment, il doit aider le "client" du concepteur à mieux définir le cahier des charges de son application en le sensibilisant aux nombreuses contraintes potentielles de celle-ci. De même, il doit automatiser le choix des structures d'AMCC en proposant une aide au dimensionnement de celles-ci (aspects interactions mutuelles entre les différents

composants, niveaux de limitation de courant...) ainsi que des évaluations qualitatives de leurs performances.

- Il doit pouvoir **traiter plusieurs solutions en même temps et en parallèle** (de façon séparée), en distinguant bien les différents raisonnements qu'il utilise pour obtenir chacune d'elles.

2) CARACTERISTIQUES DU SYSTEME EXPERT

a) Prise en charge des aspects itératifs

Le système doit prendre en charge tous les aspects itératifs [TRI91] du processus de conception que l'on peut systématiser. Ceci concerne, par exemple, des affectations de paramètres ou des choix de conception liés à des tests, avec des remises en question, jusqu'à obtention d'une solution satisfaisante.

b) Evaluation des performances

Il doit offrir des fonctions d'évaluation pondérées des performances. Par exemple, cela consiste à donner plus d'importance au critère de coût qu'à celui de rendement.

c) Bases de connaissances évolutives

L'ajout ou la suppression de connaissances issues de l'expertise doit se faire dans un formalisme simple. Il faut que tout expert puisse tester et augmenter les connaissances du système, sans être pour autant un spécialiste de la structure des programmes qui constituent ce système.

d) Couplage avec d'autres outils

Le système expert doit être compatible avec d'autres outils (par exemple : ceux de simulation d'AMCC) qu'il doit éventuellement piloter automatiquement. Par exemple, ceci peut consister à créer l'application de simulation correspondant à chaque solution choisie.

e) Génie logiciel

La structure du système expert doit être évolutive. Notamment, si on souhaite systématiser une nouvelle partie du processus de conception ou affiner des modèles de connaissance, cela doit se

faire sans remettre en cause toute l'architecture du système.

3) STRUCTURE D'UN SYSTEME EXPERT DE CONCEPTION

a) Architecture d'un système expert

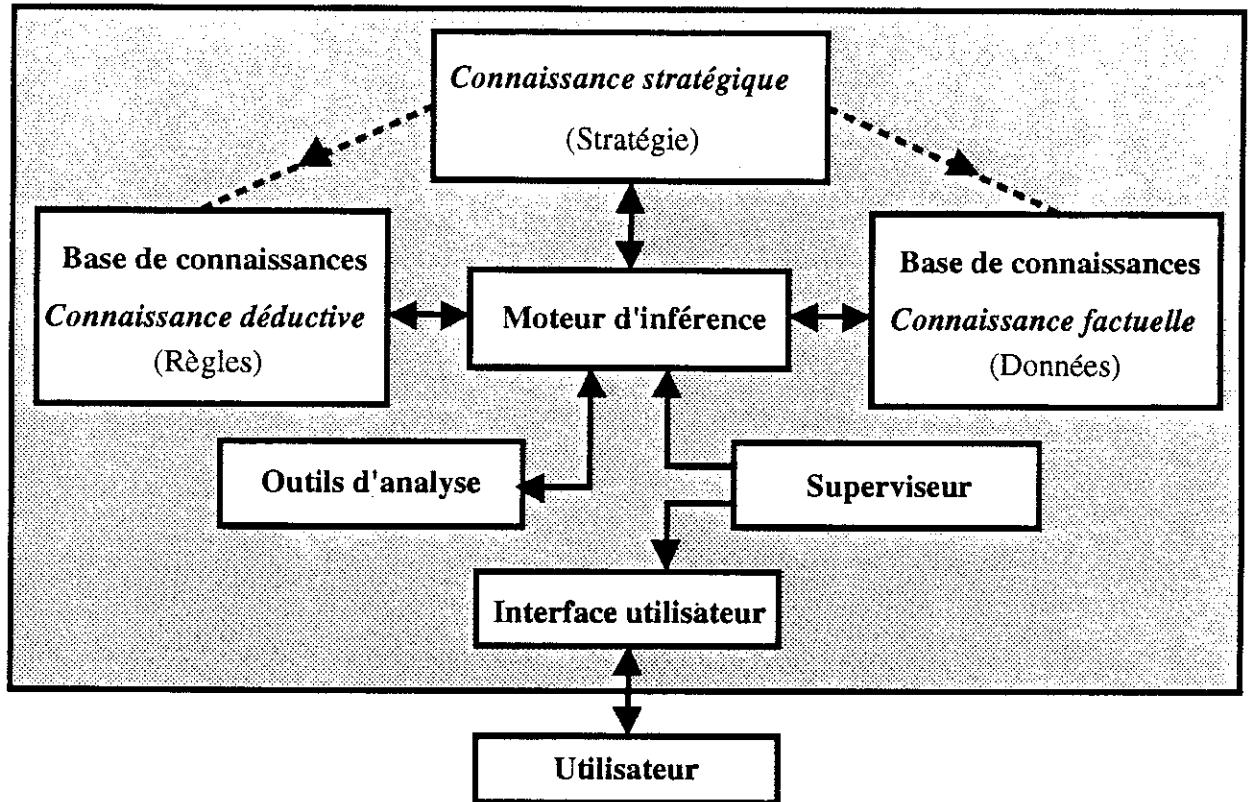


Figure I.13 : Architecture d'un système à bases de connaissance

Un système expert est souvent structuré en plusieurs modules (figure I.13). Une architecture, utilisée dans de nombreux systèmes, sépare la connaissance en différentes bases :

- la connaissance factuelle (données),
- la connaissance déductive (règles),
- la connaissance stratégique.

Un moteur d'inférence applique les règles de la connaissance déductive sur les données de la connaissance factuelle selon la stratégie de conception qui est définie dans le système expert.

Pour une bonne exécution du processus de conception, le système peut utiliser des outils d'analyse (outils de calcul, outils de simulation ...)

Un superviseur permet de gérer l'ensemble et d'assurer le lien entre le système et les interfaces

utilisateur.

Nous allons maintenant faire un balayage rapide des principaux formalismes de connaissance que nous utilisons dans notre système expert. Ceci sera développé en détail au chapitre V, pour notre application.

b) La connaissance factuelle

Elle englobe l'ensemble des données manipulées par le concepteur. Comme le nombre de ces dernières est élevé, il est conseillé de les structurer afin de faciliter leur gestion et d'assurer l'évolution du système expert. Le formalisme "**objet**" est souvent utilisé pour cette connaissance (cf chapitre V).

c) La connaissance déductive

La connaissance déductive représente le raisonnement humain dans le processus de conception, de prise de décision, de planification de diagnostic, d'interprétation des données et de compréhension du langage [HAT91].

Dans notre système, nous avons choisi un formalisme bien connu et utilisé dans de nombreux systèmes experts pour la représenter : la **règle de production**. Elle est du type "si condition alors action". Ce formalisme rigoureux et parfois contraignant, ne permet pas de modéliser finement certains aspects de la connaissance. Cependant, comme le raisonnement qui lui est associé est facile à comprendre, nous l'avons utilisé pour valider efficacement certains concepts que nous voulions mettre en oeuvre.

Remarque : Dans le cadre de la conception des AMCC, la connaissance est souvent caractérisée par des termes imprécis (faible, fort, ...) ou incertain (plutôt, préférable). Ces termes apparaissent au niveau des données qui sont floues, mais aussi au niveau des règles dont la vraisemblance est pondérée (entre impossible et certain).

Pour représenter à la fois des phénomènes complexes à modéliser et le bon sens de l'expert, des règles qualitatives peuvent être utilisées. Par exemple, on utilise de telles règles pour déduire que : "si on supprime brutalement l'alimentation de l'inducteur d'une machine à courant continu, alors celle-ci risque de se détériorer si aucune précaution n'est prise".

d) La connaissance stratégique

Elle permet d'assurer le contrôle du raisonnement. C'est généralement une modélisation

structurée de la stratégie utilisée par l'expert, mais avec toutes les contraintes techniques liées à l'architecture du système expert et au contrôle du raisonnement (condition de déclenchement des règles, modes d'exécution des règles : séquentiel ou parallèle, etc...).

Ainsi, elle permet de déterminer les critères d'application d'une règle et de modification de la connaissance factuelle (bases de données) :

- comment ?
- quand ?

Pour cela, le concepteur du système expert doit élaborer une stratégie en fonction des objectifs qu'il souhaite atteindre, par exemple :

- fournir au moins une solution si elle existe,
- fournir la meilleure solution si elle existe,
- assurer une bonne cohérence entre les différentes sources de connaissances, notamment une bonne gestion des données ainsi que de l'emploi des règles,
- aboutir le plus rapidement possible à une solution si elle existe, sinon, montrer pourquoi le problème est insoluble.

Remarque : Parfois, la stratégie ne peut pas être dissociée des données et des règles, c'est-à-dire qu'elle ne peut être contenue dans les règles.

Dans le chapitre V, nous présentons la démarche de conception que nous proposons dans notre système expert et nous définissons les différents concepts que nous mettons en oeuvre dans ce système expert pour que la stratégie de conception s'exécute correctement et que l'exploitation des résultats soit percutante.

Afin de concrétiser les idées que nous avançons, nous montrons les bases du système expert d'aide à la conception d'AMCC que nous avons réalisé, dans le chapitre VI.

Maintenant, nous allons commencer par présenter la méthodologie de simulation des AMCC que nous avons mise en oeuvre.

Chapitre II

Démarche proposée en simulation numérique

d'associations machine - convertisseur - commande

CHAPITRE II

DEMARCHE PROPOSEE EN SIMULATION NUMERIQUE D'AMCC

A partir de l'analyse qui a été faite au chapitre I, sur la simulation numérique des AMCC, nous proposons la démarche de simulation suivante.

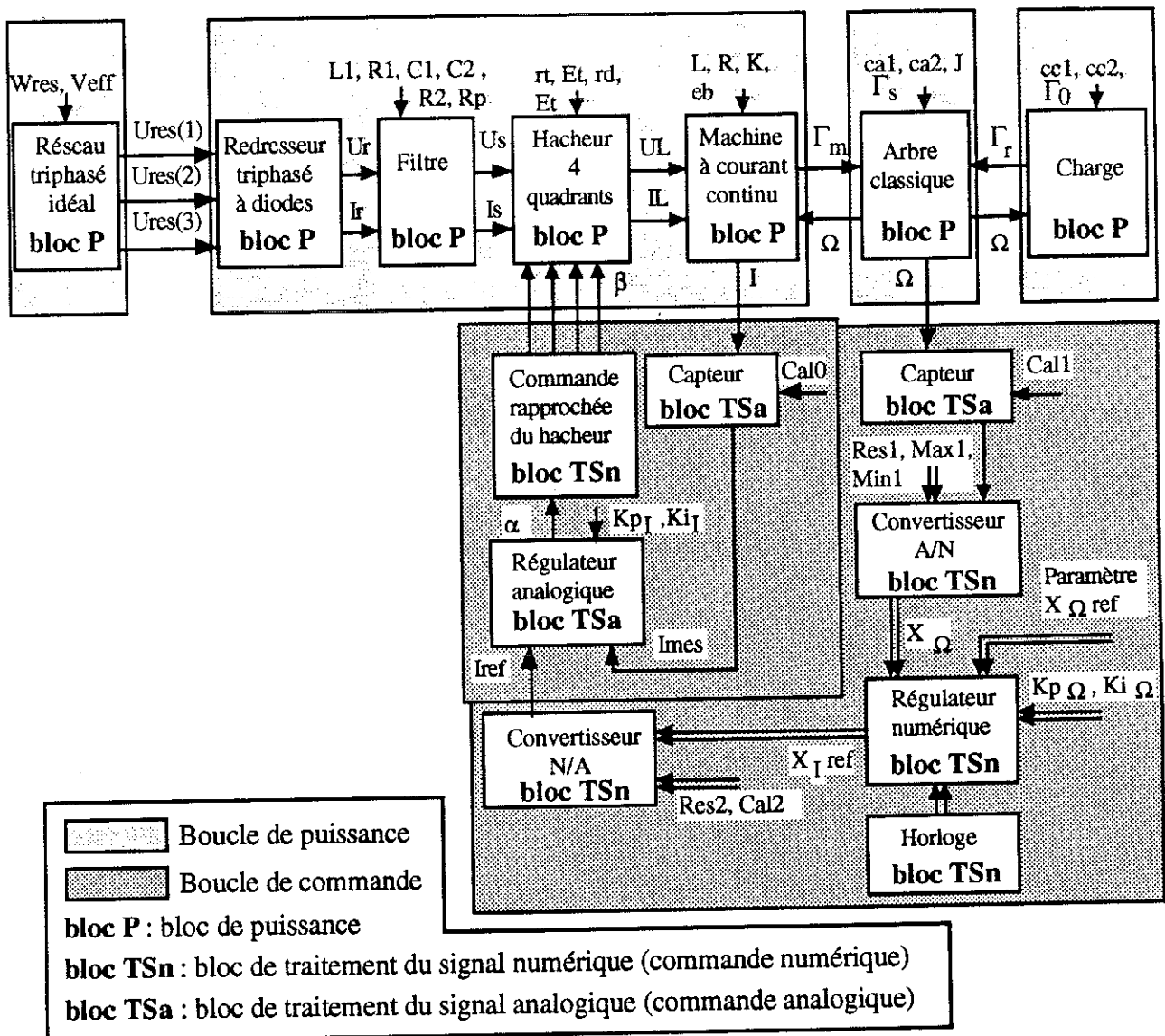


Figure II.1 : Exemple de décomposition en blocs d'une AMCC

Nous demandons à l'ingénieur, ou chercheur, qui souhaite simuler une AMCC de procéder successivement en plusieurs étapes :

- 1) Il doit décrire son AMCC sous forme de blocs, de façon à en expliciter les différents composants (cf exemple de la figure II.1).
- 2) Il doit fournir un modèle pour chaque bloc, dans le formalisme de représentation que nous lui imposons. Ce travail peut consister à récupérer :
 - soit des jeux d'équations répertoriés dans une notice d'utilisateurs ou dans un fichier informatique,
 - soit des algorithmes écrits sous forme de programmes informatiques.
- 3) Il doit fournir le séquençement de l'exécution des différents blocs de la boucle de commande numérique. Ceci correspond à l'ordre d'exécution de ces blocs.

Ensuite, nous lui proposons d'utiliser le générateur de programmes de simulation que nous avons conçu en langage de calcul formel (calcul symbolique) au cours de cette thèse. Celui-ci fournit une résolution globale des équations de puissance et de commande analogique.

Dans ce chapitre, nous présentons les différents aspects de cette méthodologie de simulation que nous proposons.

A. DEMARCHE PROPOSEE POUR L'UTILISATEUR

1) LE BLOC : UNITE DE BASE DE LA DESCRIPTION

Nous utilisons la description en blocs, pour que l'utilisateur aborde la simulation comme un banc d'essai, mais avec une plus grande flexibilité, et sans risque de destruction de composants. L'utilisateur doit ainsi pouvoir modifier aisément sa simulation et interchanger des blocs sans avoir à reformuler complètement la description de son AMCC.

a) Définition des blocs

Nous proposons de décrire les AMCC comme un ensemble d'entités physiques qui ont chacune une fonction bien définie. Nous appelons ces entités : blocs. Ces derniers peuvent représenter une machine électrique, un convertisseur statique, un régulateur, un capteur, etc... (figure II.1). Pour chaque bloc, nous définissons ses principales grandeurs qui peuvent être reliées entre elles par :

- un jeu d'équations,
- un algorithme (tests et calculs séquentiels),
- une combinaison des deux.

Pour représenter un bloc, nous proposons plusieurs formalismes de description. C'est en fonction des caractéristiques du bloc qu'on utilise un formalisme plutôt qu'un autre.

Ainsi, un bloc peut être représenté et répertorié en bibliothèque, sous la forme :

- d'équations écrites dans un fichier texte informatique, ou tout simplement, sur une feuille de papier (bien rangée !),
- d'un programme informatique qui représente l'algorithme d'une fonction de commande numérique ou des tests,
- d'une combinaison des deux, lorsque des tests doivent être utilisés en parallèle avec des équations.

Les programmes de simulation réalisés par le générateur, ne présentent aucune similitude avec la description par blocs des AMCC. Pour les parties puissance et commande analogique, l'utilisateur fournit les équations de façon désordonnée, en dehors de tout formalisme informatique, puis le générateur les réorganisent suivant des critères d'homogénéité.

L'entrée de ces équations dans le générateur est manuelle mais elle pourrait être automatisée à l'aide d'une interface. Ce travail purement informatique n'a pas encore été réalisé, car il n'est pas essentiel pour la validation de la démarche de simulation.

b) Intérêt des blocs

Une telle approche offre de nombreux intérêts :

- Une description en blocs permet de décomposer la complexité des AMCC.
- Ayant la possibilité de récupérer un travail déjà fait et stocké en bibliothèque, l'utilisateur ne se focalise que sur des aspects délicats de son AMCC et gagne un temps considérable pour développer une nouvelle application.

- La modification d'une application de simulation ne consiste alors qu'à remplacer des blocs (c'est-à-dire des équations ou des programmes élémentaires) par d'autres blocs. Notons que ce type de modification se fait au niveau de la description en "blocs" des AMCC. Ceci peut donc conduire à une refonte totale du programme de simulation final, mais c'est le générateur qui réalise ce travail.

2) METHODOLOGIE DE DESCRIPTION EN BLOCS D'UNE AMCC

a) Démarche proposée

Pour que l'utilisateur puisse bénéficier d'un formalisme générique pour décrire ses blocs, nous lui proposons une démarche qui résulte du fruit de l'expérience acquise au sein du laboratoire, sur la simulation numérique d'AMCC (figure II.2). Il doit alors :

- Décomposer son application en plusieurs groupes qui ont la même fonction (puissance ou commande) et la même structure (traitement de grandeurs discrètes ou de grandeurs continues), et que nous appelons boucles (§ a.1).
- Décomposer ces boucles en blocs (§ a.2).

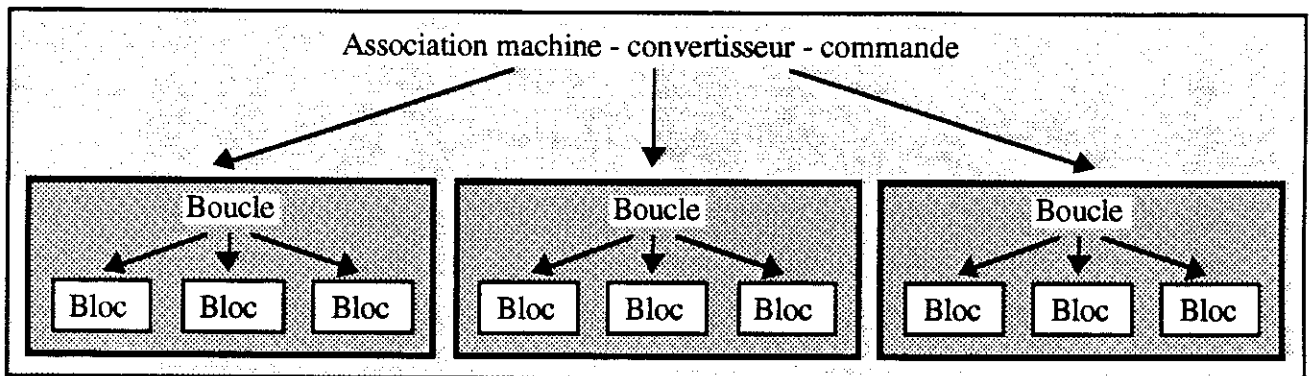


Figure II.2 : Décomposition en blocs d'une AMCC

a.1 Décomposition en boucles

Cette étape à plusieurs finalités :

- Améliorer, en traitant des problèmes plus petits, les traitements numériques de la simulation et les opérations faites par le générateur.
- Distinguer les parties puissance et commande.

- Faire apparaître les parties analogiques ou contenant des grandeurs continues, et les parties numériques contenant des grandeurs discrètes.

Dans une AMCC, nous distinguons plusieurs boucles (figure II.3).

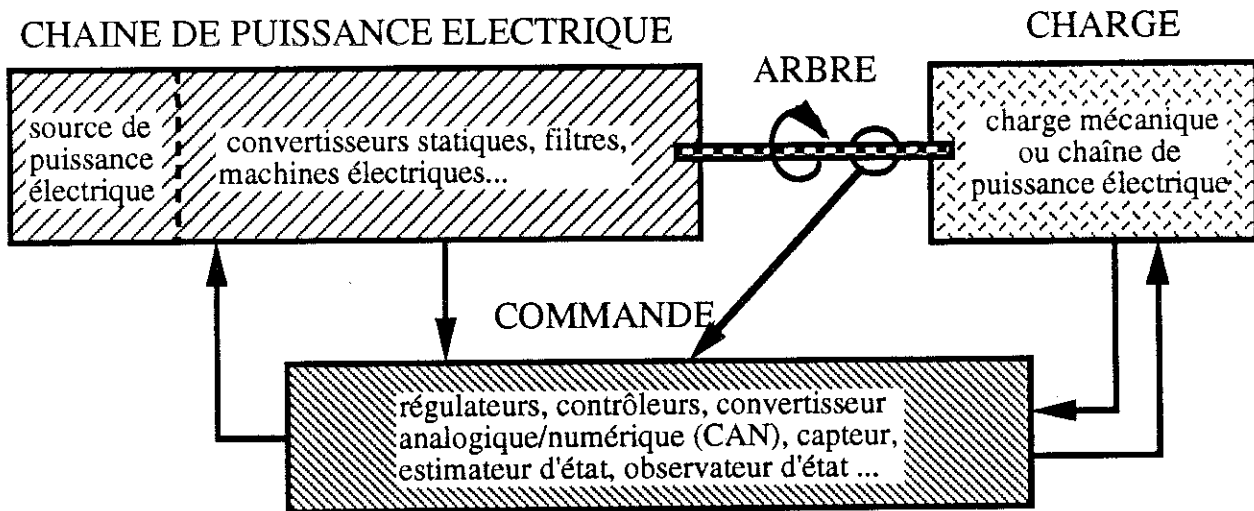


Figure II.3 : Exemple général d'AMCC

- Les boucles de puissance qui sont :

- * Soit **mécaniques**, c'est à dire qui correspondent à des charges et/ou à des couplages mécaniques directs ou indirects, par exemple : des engrenages, des réducteurs, des systèmes poulies-courroies ... Elles sont caractérisées par des inerties, des frottements, des élasticités et des jeux.
- * Soit **électriques**, c'est à dire qui contiennent des blocs source de puissance, filtre, machine électrique, convertisseur statique, réseau électrique...

Les boucles de puissance sont représentées par des équations qui interagissent entre elles, et éventuellement, par des tests (par exemple, lorsqu'il est nécessaire de modéliser les commutations d'interrupteurs de puissance).

- Les **arbres des moteurs** et les **sources de tension** qui lient plusieurs boucles de puissance entre elles. Ces éléments sont à la fois des boucles et des blocs de puissance. Nous les distinguons des boucles précédentes, à des fins d'optimisation des traitements faits par le générateur et des programmes de simulation (cf paragraphe II.D).

- La **commande** qui assure le contrôle des boucles de puissance. Elle peut être réalisée selon trois technologies différentes qui peuvent être combinées entre elles :

- * en numérique programmée (informatique industrielle),
- * en numérique cablée (électronique logique),
- * en analogique (électronique analogique).

Une combinaison de ces différentes technologies est souvent utilisée. Par exemple, il est fréquent, pour des raisons de rapidité, d'utiliser des boucles de contrôle de courant analogique, et, pour des raisons de convivialité, d'utiliser une boucle de vitesse numérique.

a.1 Décomposition en blocs

Dans chaque boucle, nous définissons plusieurs **types de blocs** que nous distinguons par leur structure et leur fonction (figure II.4) :

- les blocs de puissance : **Blocs P**,
- les blocs de traitement du signal analogique (commande analogique) : **Blocs TSa**,
- les blocs de traitement du signal numérique (commande numérique) : **Blocs TSn**.

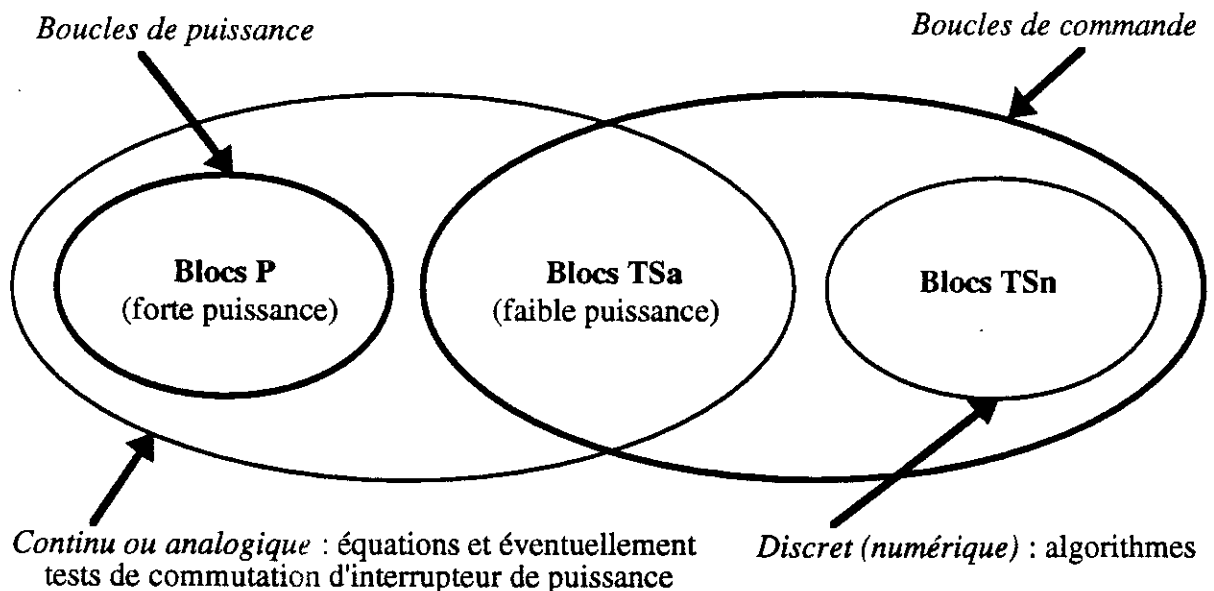


Figure II.4 : Classement des blocs en fonction de leur structure et de leur fonction

Comme une commande analogique peut être substituée ou combinée avec une commande numérique, nous distinguons, pour une clarté d'esprit, les blocs TSa et P, bien qu'ils aient le même formalisme de représentation et qu'ils ne diffèrent essentiellement que par leur niveau de puissance.

b) Caractéristiques d'un bloc

Maintenant, nous pouvons définir précisément un bloc. Pour nous, il est caractérisé par :

- son type (P, TSn, TSa),
- sa fonction (régulateur, convertisseur,...),
- son niveau de modélisation (moyen [MID76], à topologie fixe ou variable (cf chapitre I),
- différentes grandeurs (variables d'état, paramètres physiques, consignes de contrôle...),
- des équations ou relations entre ces grandeurs,
- des tests (événements, tops d'horloge, ordres de commutation d'interrupteur, etc.).

Nous allons maintenant présenter comment sont décrites et résolues la partie commande numérique et les parties commande analogique et puissance.

B. TRAITEMENT DE LA PARTIE COMMANDE NUMERIQUE

1) ASPECTS A SIMULER

La commande numérique présente certaines spécificités qui peuvent modifier, de façon notable, le comportement de l'ensemble du système contrôlé. Ces spécificités sont :

- La façon dont est implantée la commande dans le calculateur du banc d'essai. Ceci fait intervenir des problèmes complexes de gestion de tâches (avec des priorités, des interruptions...) pour les différents modes de fonctionnement de la commande [FAR91] [PET92].
- La durée d'exécution de chaque fonction, qui conditionne la valeur de sa période d'échantillonnage et l'enchaînement avec lequel l'ensemble de ces fonctions s'exécute.
- La période d'échantillonnage des fonctions et le format des grandeurs numériques qui sont manipulées par cette commande, et qui ont une influence sur les performances des régulations et sur le comportement global de l'AMCC.

2) DESCRIPTION DES BLOCS TS_n

Pour modéliser, simplement et de façon générique, ces différents aspects, nous avons élaboré un formalisme général de représentation des blocs TS_n (cf figure II.5 et annexe VI).

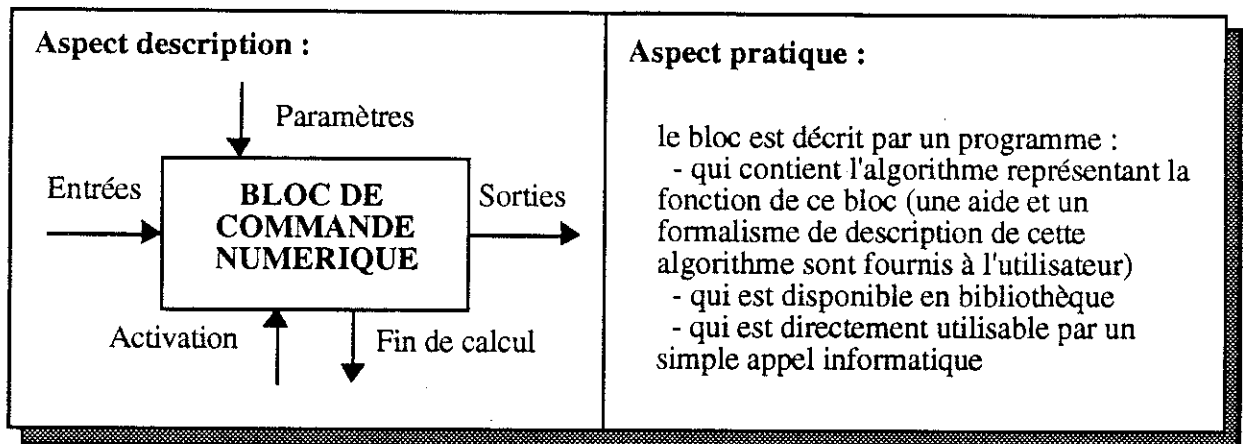


Figure II.5 : Bloc de commande numérique (TS_n)

Pour prendre en compte la durée et la période d'exécution des blocs, nous avons défini les grandeurs "activation" et "fin de calcul". Les sorties sont calculées à partir des entrées, grâce à un algorithme. En annexe VI, nous présentons en détail, comment nous prenons en compte le format des données et la gestion des tâches dans le calculateur.

3) RESOLUTION SEQUENTIELLE DES BLOCS

De par sa structure, la commande numérique est caractérisée par ses périodes d'échantillonnage et la séquentialité de l'exécution de ses fonctions.

Cette dernière se retrouve au niveau de la simulation. Tous les blocs sont des programmes élémentaires qui représentent chacun une fonction de la commande. Au cours de la simulation, l'exécution de ces blocs se fait séquentiellement.

Le séquençage avec lequel ces blocs s'exécutent est propre à chaque commande. Nous proposons à l'utilisateur de le formuler à l'aide d'un algorithme qui gère les "activations" et les "fins de calcul" des blocs. Ceci est physiquement représenté par un bloc "séquenceur". Cependant, pour les cas où un seul mode de commande est utilisé, on peut employer des blocs "horloges" (cas de la figure II.6).

L'enchaînement des blocs se fait par la connexion de leurs entrées / sorties respectives. Ceci correspond à écrire les appels informatiques successifs des programmes qui représentent ces blocs. L'exécution de ces derniers, n'est autorisée que par leur activation, qui est généralement affectée par le séquençement.

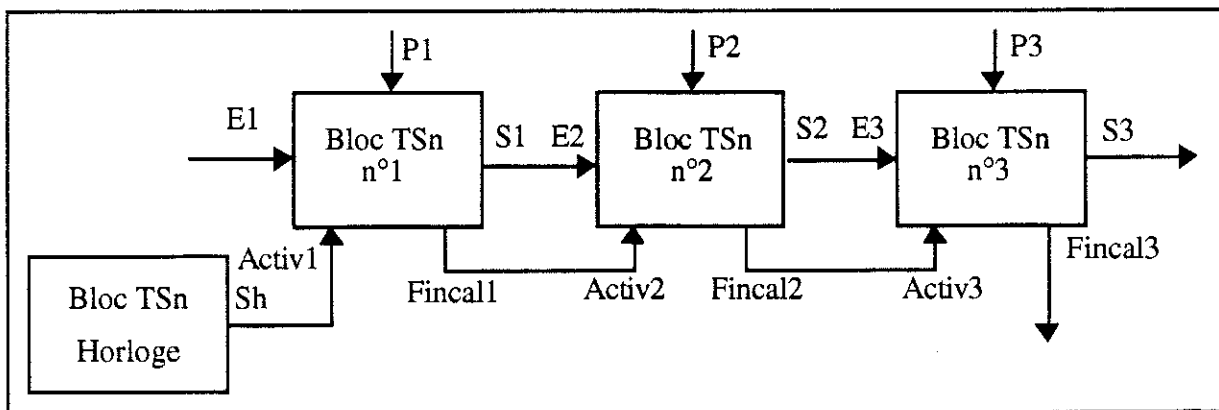


Figure II.6 : Exemple d'enchaînement séquentiel de trois blocs TSn

La part de programmation informatique à la charge de l'utilisateur reste faible si les blocs nécessaires pour décrire l'application sont disponibles en bibliothèque. Ce travail correspond, pour l'ensemble des blocs TSn, à l'écriture de leur séquençement (affectation des activations) et de leur enchaînement (appel informatique des programmes).

Remarque : Cette programmation se termine par l'appel informatique des programmes de gestion des configurations de chaque bloc multiconfiguration (convertisseurs statiques...)

C. TRAITEMENT DE LA PARTIE PUISSANCE ET COMMANDE ANALOGIQUE

Comme elles sont caractérisées par des équations continues, la puissance et la commande analogique sont traitées ensemble.

1) ASPECTS A SIMULER

a) La commande analogique

Elle correspond à des circuits d'électronique analogique. Pour un mode de fonctionnement donné, elle peut être activée ou non, par une validation.

b) La puissance

Les blocs de puissance sont essentiellement caractérisés par des équations continues. Elles peuvent être simples (cas d'un filtre de courant) mais aussi très complexes (cas de la machine synchrone présentée en annexe V).

Certains de ces blocs contiennent des interrupteurs de puissance (diodes, transistors, thyristors...). Ceci concerne plus particulièrement les convertisseurs statiques. Par le jeu d'ouverture et de fermeture de ces interrupteurs, certaines grandeurs sont reliées entre elles ou annulées. Cela entraîne une évolution de la taille du système d'état global de la boucle de puissance considérée.

On peut représenter un tel bloc par deux types de modèle :

- le modèle à **topologie variable** : un jeu d'équations simples par configuration du bloc, et des tests qui gèrent l'enchaînement dynamique de ces jeux d'équations, en cours de simulation,
- le modèle à **topologie fixe** : un seul jeu d'équations paramétrées pour toutes les configurations du bloc, et des tests qui gèrent l'affectation dynamique des paramètres de ce jeu d'équations, en cours de simulation.

Comme cela a été signalé au chapitre I, on a souvent recours au premier type de modèle pour la simulation numérique des AMCC, mais dans notre démarche, il est possible d'utiliser l'un ou l'autre de ces modèles.

2) DESCRIPTION DES BLOCS

a) Description des blocs TSa

Nous représentons ces blocs par (figure II.7) :

- des équations paramétrées (paramètres physiques),
- des grandeurs échangées avec d'autres blocs, et liées entre elles par les équations précédentes,
- une validation.

Selon la valeur de la validation, la fonction du bloc est active ou inactive. Pour le générateur, cela consiste à créer deux configurations pour ce bloc :

- une vide où rien n'est exécuté : inactive,
- une avec les équations décrites précédemment : active.

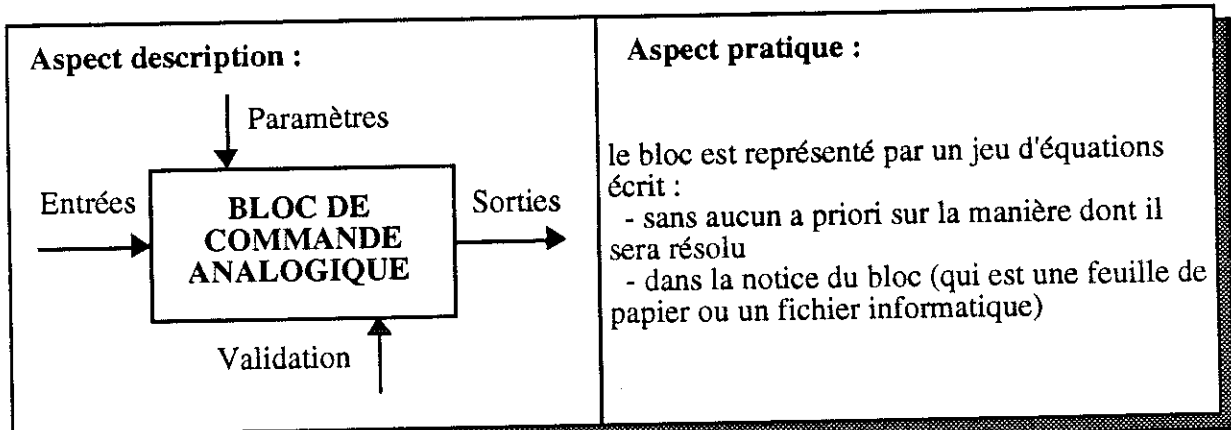


Figure II.7 : Bloc de commande numérique (TSa)

Remarque : Pour les blocs de commande analogique, la notion d'entrée / sortie est purement fonctionnelle. Elle correspond directement aux entrées/sorties des blocs TSn qui ont la même fonction. Par exemple, un régulateur analogique (ou numérique) a :

- en entrée : une consigne à respecter et une grandeur qui doit être asservie à cette consigne,
- en sortie : une grandeur de commande qui peut être destinée à la commande rapprochée d'un convertisseur statique d'une boucle de puissance.

Pour les blocs de commande analogique, les entrées / sorties fonctionnelles ne sont que des grandeurs échangées (cf paragraphe II.C.2.b).

b) Description des blocs P

Nous représentons un bloc de puissance par (figure II.8) :

- un ou plusieurs jeux d'équations paramétrées et écrites de façon désordonnée (sans a priori sur la façon dont elles sont résolues),
- des grandeurs échangées avec d'autres blocs et liées entre elles par les équations précédentes,

- si ce bloc contient des interrupteurs :
 - une grandeur de commande,
 - des tests qui sont écrits par l'utilisateur dans un programme informatique, et qui gèrent l'enchaînement des configurations de ce bloc.

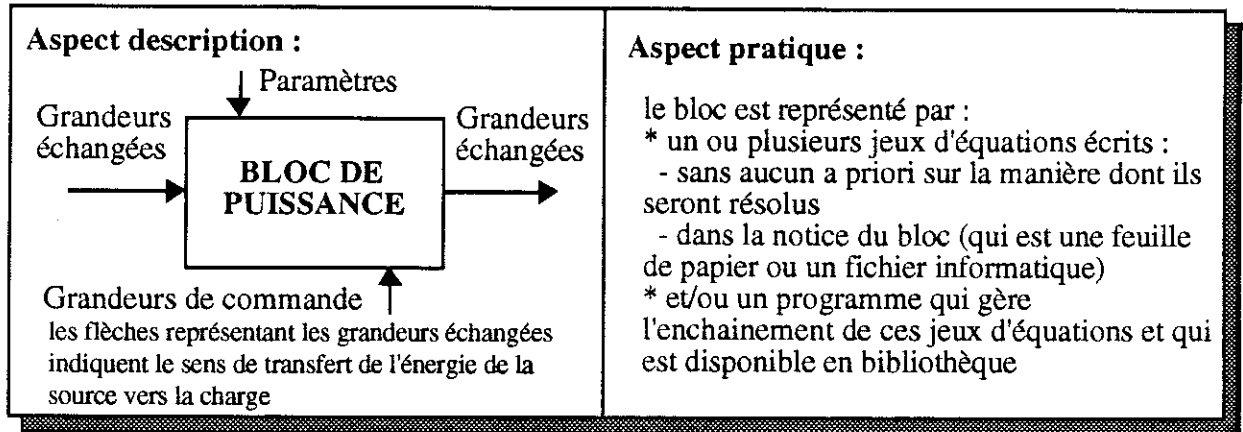


Figure II.8 : Bloc de puissance (P)

Les équations des blocs P et TSa peuvent être fournies de façon séparée et désordonnée, elles sont résolues de façon globale comme nous allons le présenter maintenant.

3) RESOLUTION GLOBALE DES EQUATIONS

Par rapport aux approches classiques utilisées en simulation numérique d'AMCC, notre approche propose une résolution globale des équations de puissance et de commande analogique. Pour cela, nous avons réalisé un générateur d'équations et de programmes de simulation. A partir des équations décrites de façon formelle pour chaque bloc, par l'utilisateur, notre générateur construit automatiquement un système d'équations global (pour l'ensemble des blocs) qui est homogène, réduit, simplifié et concrètement possible. Il génère ensuite les programmes de simulation correspondants.

a) Le générateur

Pour atteindre ces performances et pour assurer une évolution à notre générateur, nous l'avons réalisé dans un langage de calcul formel (symbolique) : Macsyma. C'est un langage de haut niveau qui nous a permis d'obtenir un générateur de taille acceptable, simple, facilement modifiable, adaptable à plusieurs outils de simulation d'AMCC existants, et compatible avec d'autres outils d'analyse et d'aide (systèmes experts).

Un langage de programmation classique (C, pascal...) aurait pu être utilisé. Cependant, pour obtenir des performances équivalentes, cela aurait nécessité un travail de programmation informatique beaucoup plus important et complexe, et cela aurait considérablement alourdi la structure de l'outil de simulation final.

b) Formalisme de représentation des équations

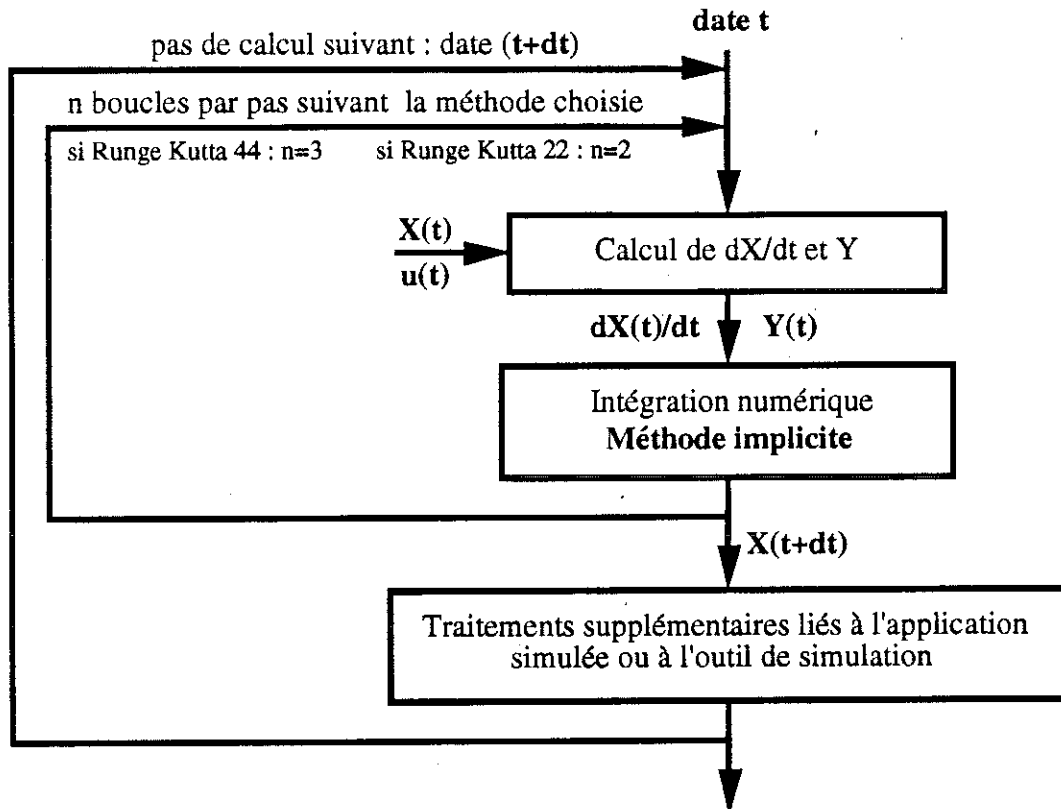


Figure II.9 : Résolution dynamique

Le générateur procède en plusieurs étapes. D'abord, pour chaque configuration d'une partie puissance et commande analogique de l'AMCC étudiée, il rassemble l'ensemble des équations des blocs, pour donner une équation d'état global du type :

$$f(\dot{X}(t), X(t), Y(t), u(t), \theta(t), t) = 0 \quad (1)$$

où :

- . t : temps courant,

- . $X(t) = [x_i]_n$: vecteur d'état à l'instant t , de dimension n ,

- . $\dot{X}(t) = dX(t)/dt = [\dot{x}_i]_n$: dérivée temporelle de $X(t)$ à l'instant t ,

- . $Y(t) = [y_i]_m$: vecteur de sortie à l'instant t , de dimension m ,

- . $u(t) = [u_i]_r$: vecteur d'entrée (excitation) à l'instant t , de dimension r ,

- . $\theta(t) = [\theta_i]_s$: vecteur des perturbations ou grandeurs provenant d'une autre partie de l'AMCC étudiée. (qui peuvent être des grandeurs d'état).

Le générateur analyse et reformule alors cette équation différentielle de façon à ce qu'elle puisse être intégrée par un module de résolution implicite d'équations différentielles (figure II.9). Ceci consiste à extrapoler symboliquement $\dot{X}(t)$ et $\dot{Y}(t)$ en fonction de $X(t)$, $\theta(t)$, $u(t)$ et t .

Le module de résolution implicite fournit $X(t)$ et les variables d'état de $\theta(t)$, et les autres boucles de puissance et commande analogique fournissent les autres composantes de $\theta(t)$.

c) Réduction de l'équation d'état globale

Parmi les éléments qui interviennent dans une boucle de puissance, les blocs multi-configuration, dont les convertisseurs sont modélisés par topologie variable (figure II.10), présentent des aspects propres qui sont gérés automatiquement par notre générateur.

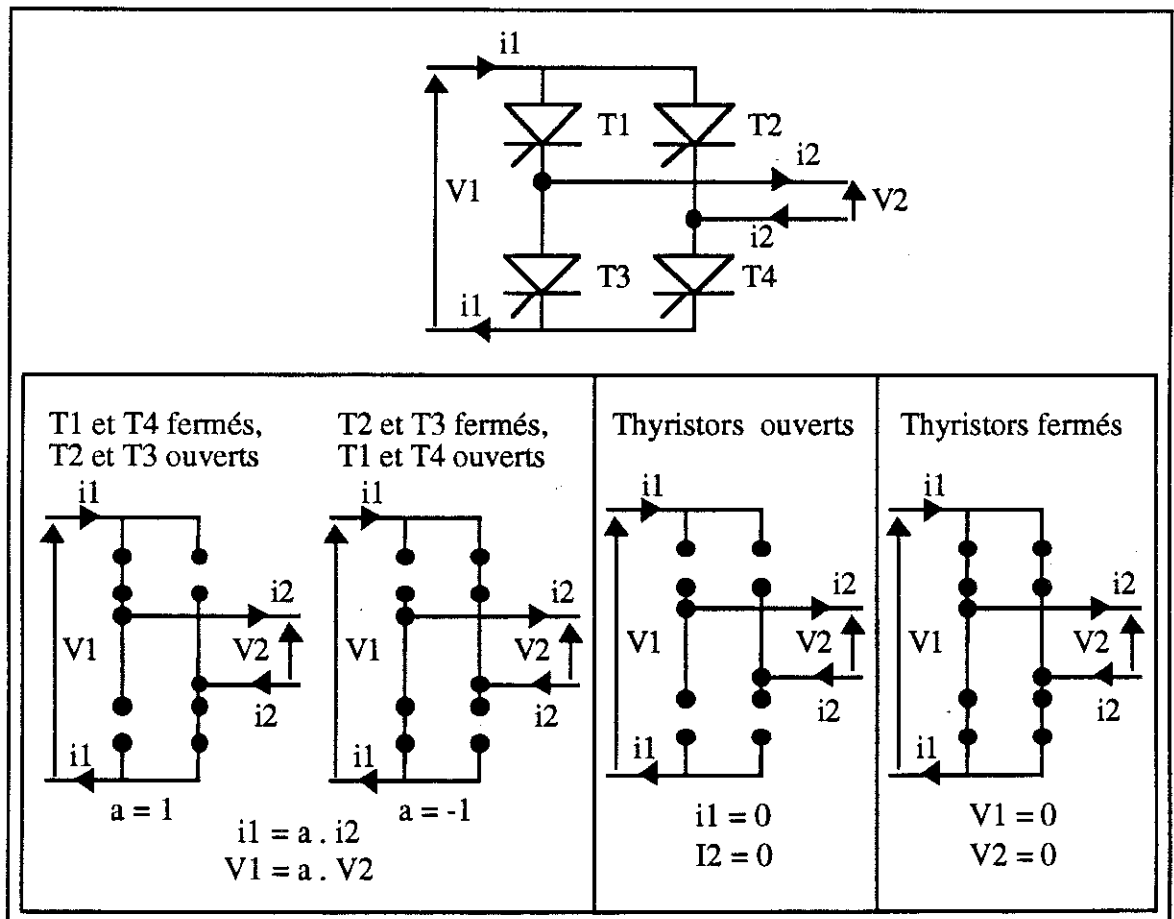


Figure II.10 : Exemple de modèle à topologie variable d'un commutateur de courant monophasé

En effet, suivant les différentes configurations qu'ils prennent, par le jeu d'ouverture et de fermeture des interrupteurs qui les composent, certaines grandeurs sont reliées entre elles ou sont annulées.

Les équations qui régissent un tel bloc, sont souvent très simples et de la forme :

$$\sum_{i=1}^n a_i \cdot V_i = 0$$

où les a_i sont des coefficients algébriques (non physiques) qui valent généralement ± 1 , et où les V_i sont les grandeurs échangées par le bloc avec les autres blocs.

Notons qu'une relation du type

$$\sum_{i=1}^n a_i \cdot V_i = 0$$

doit aussi être interprétée comme

$$\sum_{i=1}^n a_i \cdot \frac{d^k V_i}{dt^k} = 0 \text{ où } k \in \mathbb{N}^*$$

qui est la relation entre les dérivées $k^{\text{ièmes}}$ des V_i .

En raison de ces combinaisons linéaires entre grandeurs, la dimension de l'équation d'état globale (dimension de $X(t)$), évolue selon les configurations des convertisseurs statiques ou de tous les blocs contenant au moins un interrupteur.

Ainsi, lors de la réorganisation, le générateur réduit le vecteur $X(t)$ en un vecteur $X_1(t)$, qui est obtenu par une combinaison linéaire des composantes de $X(t)$. Il en résulte l'équation d'état globale réduite et reformulée par :

$$f(\dot{X}_1(t), X_1(t), Y_1(t), Y_2(t), u(t), \theta(t), t) = 0 \quad (2)$$

où : $X_1(t) = [x_{1i}]_{n_1}$: vecteur d'état à l'instant t , de dimension $n_1 (\leq n)$,

$$\forall i, 1 \leq i \leq n_1 : x_{1i} = \sum_{j=1}^n \alpha_{ij} \cdot x_j \text{ (en général, } \alpha_{ij} = 1 \text{ ou } -1 \text{ ou } 0),$$

$$\dot{X}_1(t) = dX_1(t)/dt = [\dot{x}_{1i}]_{n_1} : \text{dérivée temporelle de } X_1(t),$$

$$Y_1(t) = [y_{1i}]_{m_1} : \text{vecteur de sortie réduit, de dimension } m_1 (\leq m),$$

$$\forall i, 1 \leq i \leq m_1 : y_{1i} = \sum_{j=1}^m \beta_{ij} \cdot y_j \text{ (en général, } \beta_{ij} = 1 \text{ ou } -1 \text{ ou } 0),$$

$$Y_2(t) = [y_{2i}]_{m_2} : \text{complément de } X_1(t) \text{ et } Y_1(t) \text{ par rapport à } X(t) \text{ et } Y(t) ; \text{ vecteur de dimension } m_2 = (n + m) - (n_1 + m_1),$$

$$\forall i, 1 \leq i \leq m_2, y_{2i} = \sum_{j=1}^n \mu_{ij} \cdot x_j + \sum_{k=1}^m \lambda_{ik} \cdot y_k$$

$$\text{(en général, } \lambda_{ik} = 1 \text{ ou } -1 \text{ ou } 0, \text{ et } \mu_{ij} = 1 \text{ ou } -1 \text{ ou } 0).$$

d) Formes de réorganisation

Cette équation réduite (2) peut être écrite sous trois formes différentes.

d.1) Forme classique des équations d'état : forme [1]

$$\begin{aligned}
 \dot{x}_{1i} &= \sum_{j=1}^{n_1} a_{ij}(\theta).x_{1j} + \sum_{j=1}^r b_{ij}(\theta).u_j, \quad 1 \leq i \leq n_1 \\
 [1] \quad y_{1i} &= \sum_{j=1}^{n_1} c_{ij}(\theta).x_{1j} + \sum_{j=1}^r d_{ij}(\theta).u_j, \quad 1 \leq i \leq m_1 \\
 y_{2i} &= g_i(X_1, Y_1, \theta, u), \quad 1 \leq i \leq m_2
 \end{aligned}$$

où les a_{ij} , b_{ij} , c_{ij} , d_{ij} sont des fonctions linéaires ou non des composantes de $\theta(t)$. Les g_i sont des fonctions linéaires ou non des composantes de $X_1(t)$, $Y_1(t)$, $\theta(t)$ et $u(t)$.

d.2) Forme triangularisée : forme [2]

C'est une forme moins explicite que [1]

$$\begin{aligned}
 \dot{x}_{1i} &= \sum_{j=1}^{i-1} \alpha'_{ij}(\theta).\dot{x}_{1j} + \sum_{j=1}^{n_1} a_{ij}(\theta).x_{1j} + \sum_{j=1}^r b_{ij}(\theta).u_j, \quad 1 \leq i \leq n_1 \\
 [2] \quad y_{1i} &= \sum_{j=1}^{n_1} \beta'_{ij}(\theta).\dot{x}_{1j} + \sum_{j=1}^{i-1} \gamma'_{ij}(\theta).y_{1j} + \sum_{j=1}^{n_1} c_{ij}(\theta).x_{1j} + \sum_{j=1}^r d_{ij}(\theta).u_j, \quad 1 \leq i \leq m_1 \\
 y_{2i} &= g_i(X_1, Y_1, \theta, u), \quad 1 \leq i \leq m_2
 \end{aligned}$$

où les a_{ij} , b_{ij} , c_{ij} , d_{ij} , α'_{ij} , β'_{ij} , γ'_{ij} sont des fonctions linéaires ou non des composantes de $\theta(t)$. Les g_i sont des fonctions linéaires ou non des composantes de $X_1(t)$, $Y_1(t)$, $\theta(t)$ et $u(t)$.

L'exploitation de l'équation d'état sous les formes [1] ou [2] en vue de sa résolution dynamique se fait par une séquence de transformations qui est facile à élaborer par notre générateur.

Par exemple, l'équation suivante mise sous la forme [1] :

$$\begin{aligned} \text{d}u/\text{d}t &= i/c \\ \text{d}i/\text{d}t &= -u/L - r \cdot i/L + E/L \end{aligned}$$

est traduite en Fortran par :

$$\begin{aligned} \text{d}u/\text{d}t &= i/c \\ \text{d}i/\text{d}t &= (E - r * i - u)/L \end{aligned}$$

A chaque pas de calcul, les valeurs de $\dot{X}_1(t)$, $Y_1(t)$ et $Y_2(t)$ sont facilement obtenues par un calcul séquentiel, tel qu'il est explicité par ces formes [1] et [2].

d.3 Forme numérique : forme [3]

C'est la forme la moins explicite que nous proposons. Elle fournit $\dot{X}_1(t)$ et $Y_1(t)$ par une résolution numérique de système d'équations linéaires :

Notre générateur reformule symboliquement l'équation d'état réduite sous la forme :

$$A(\theta) \cdot \begin{bmatrix} \dot{X}_1 \\ Y_1 \end{bmatrix} = B(X_1, \theta, u)$$

+ "appel informatique" d'un sous-programme de résolution numérique de système linéaire pour obtenir les valeurs de $\dot{X}_1(t)$ et $Y_1(t)$ à partir de celles de $X_1(t)$, $\theta(t)$ et $u(t)$,
 $y_{2i} = g_i(X_1, Y_1, \theta, u)$, $1 \leq i \leq m_2$.

où :

- $A(\theta) = [a_{ij}]_{n_1 + m_1, n_1 + m_1}$: matrice carrée de dimension $n_1 + m_1$, dont les composantes sont des fonctions linéaires ou non de celles de $\theta(t)$,
- $B(X_1, \theta, u) = [b_i]_{n_1 + m_1}$: vecteur de dimension $n_1 + m_1$, dont les composantes sont des fonctions linéaires ou non de celles de $X_1(t)$, $\theta(t)$ et $u(t)$,
- les g_i sont des fonctions linéaires ou non des composantes de $X_1(t)$, $Y_1(t)$, $\theta(t)$ et $u(t)$.

e) Classement de ces formes de réorganisation

Les trois formes de réorganisation présentées ci-dessus peuvent être classées suivant différents critères, comme le montre le tableau de la figure II.11.

Avant de développer ces critères, définissons la distinction que nous faisons entre les "petits" et "gros" problèmes. Nous disons qu'un problème est "gros", lorsque notre générateur donne des équations réorganisées, réduites et simplifiées sous les formes [1] et [2] qui font plusieurs centaines de lignes dans le format de programmation Fortran. Maintenant, nous analysons les différents critères.

Fiabilité de la simulation si "gros" problème	On ne peut rien dire (1) (2) (3) Assurée
Durée d'exécution du générateur	Faible (3) (2) (1) Elevée
Durée de simulation	Faible (2) (1) (3) Elevée
Taille mémoire occupée par les jeux d'équations réorganisés si "gros" problème	Faible (3) (2) (1) Elevée
Taille mémoire occupée par les jeux d'équations réorganisés si "petit" problème	Faible (2) (1) (3) Elevée
Exploitation de la réorganisation des équations	Faible (3) (2) (1) Elevée

Figure II.11 : Classement des formes de réorganisation des équations

e.1 Fiabilité de la simulation

Pour la simulation dynamique des "petits" problèmes, les trois formes sont fiables. Par contre, pour les "gros" problèmes, nous recommandons la forme [3].

En effet, bien que les formes [1] et [2], obtenues symboliquement par notre générateur, donnent des solutions exactes (non approchées par rapport aux méthodes numériques), elles peuvent s'avérer non optimisées pour des calculs numériques, lorsque les expressions sont complexes et très imposantes (cf annexe II.2). Par exemple, des petits nombres risquent d'être négligés de

façon incontrôlée devant des grands nombres (en raison de la précision des calculs), ce qui peut donner des résultats faux. A l'opposé, la forme [3], qui est conçue et optimisée pour être purement numérique, ne présente pas ce type d'inconvénient. Elle donnera toujours des résultats corrects.

e.2 Durée d'exécution du générateur

La forme [3] est celle qui est obtenue le plus rapidement, puis c'est la forme [2] et enfin la forme [1]. Comme cette forme [3] est la plus fiable, nous la recommandons pour tester rapidement la validité d'un modèle d'AMCC en simulation.

e.3 Durée de simulation

Pour les "petits" problèmes, nous recommandons la forme [2], puis la forme [1], car elles permettent d'optimiser les temps de simulations par rapport à la forme [3].

En effet, dans ce cas, les équations sont simplifiées, réduites et les calculs algébriques permettant d'obtenir $\dot{X}_1(t)$, $Y_1(t)$ et $Y_2(t)$ sont bien explicités. Ces calculs sont donc plus rapides qu'avec la forme [3], où le nombre d'itérations et d'opérations peut être élevé. Ce dernier est fonction du cube de la dimension de $A(\theta)$.

Ainsi, le problème traité au chapitre IV.B nécessite la forme [3], car c'est un gros problème, par contre, le cas traité au chapitre IV.A est plutôt organisé sous la forme [1].

e.4 Taille mémoire occupée par le code généré

Pour les "gros" problèmes, c'est la forme [3] qui donne généralement les programmes de taille la plus faible, alors que pour les "petits" problèmes, ce sont les formes [2] puis [1].

e.5 Exploitation de la réorganisation des équations

Comme nous le développons dans le chapitre III.D, alors que les formes [2] et [3] sont surtout intéressantes pour la simulation, la forme [1] peut être utilisée pour réaliser des analyses complémentaires :

- application de théories de l'automatique,
- élaboration de nouveaux modèles,
- etc...

e.6 Conseil d'utilisation

Nous laissons à la charge de l'utilisateur, le choix de la forme de réorganisation que doit réaliser le générateur. Ce choix devient performant avec l'habitude. Cependant, nous recommandons à l'utilisateur de valider expérimentalement ses équations en simulation, avec la forme [3], avant d'utiliser la forme [1] et éventuellement la forme [2] qui est, en moyenne, la meilleure en durée de simulation.

4) GESTION DE L'ENCHAINEMENT DES CONFIGURATIONS

Chaque bloc multiconfiguration possède son propre programme pour gérer l'enchaînement dynamique de ses configurations. Ce programme est écrit par l'utilisateur, lors de la description du bloc.

Pour une partie de puissance et de commande analogique, notre générateur récupère l'ensemble des programmes de gestion des configurations de cette partie. Ces dernières correspondent à des combinaisons des configurations des blocs. Ainsi, à partir d'une gestion par bloc, notre générateur engendre automatiquement une gestion globale de l'enchaînement des configurations de l'AMCC étudiée.

D. DECOUPLAGE ENTRE BOUCLES

1) OBJECTIFS

Afin d'optimiser le temps d'exécution du générateur et des programmes de simulation qu'il réalise, nous recommandons de découpler certaines boucles de puissance et / ou de commande analogique.

Nous pouvons faire ceci de deux façons :

- découplage par les blocs,
- découplage par les constantes de temps.

2) DECOUPLAGE PAR LES BLOCS

Les blocs permettant de découpler des boucles de puissance sont essentiellement :

- les arbres des moteurs,
- les blocs qui ne font intervenir que des grandeurs constantes ou qui dépendent uniquement du temps courant. Ce sont, par exemple, les sources de tension modélisées sans effet inductif ni capacitif.

Les boucles de puissance, qui ne sont liées entre elles que par des blocs de ce type, peuvent être considérées séparément pour leur traitement par le générateur et pour la simulation.

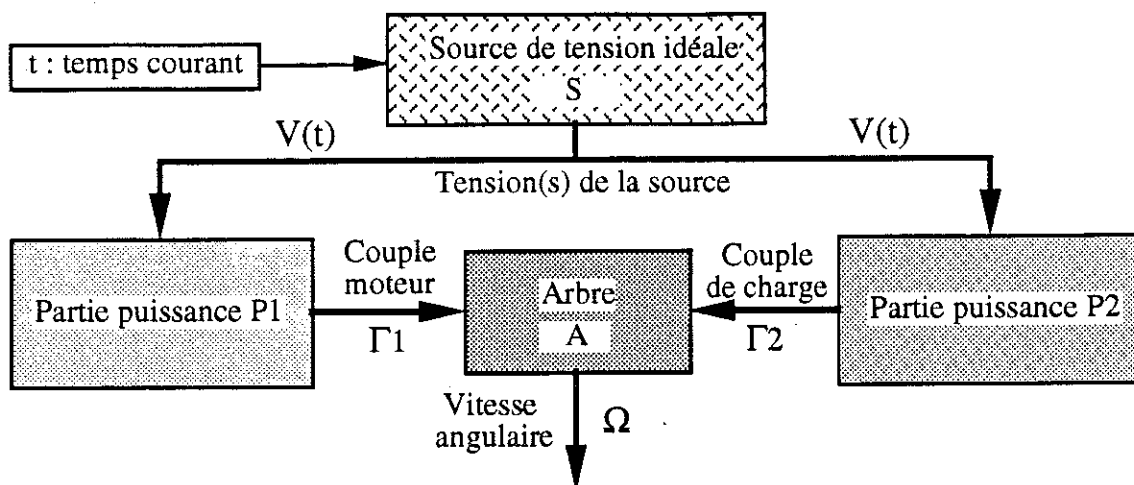


Figure II.12 : Couplage direct par un arbre de deux parties puissance indépendantes vis à vis de la source de tension et de l'arbre

Dans l'exemple de la figure II.12, les boucles de puissance P1 et P2 sont reliées entre elles par la source de tension S et par l'arbre A. Ainsi, les boucles S, P1, P2 et A peuvent être traitées séparément par notre générateur. En cours de simulation numérique, à chaque pas de calcul, elles seront résolues séquentiellement, dans l'ordre (S, P1, P2, A) ou (S, P2, P1, A).

En annexe VII, nous développons ce type de découplage. La réalisation de ce dernier n'est pas obligatoire, mais cela évite d'engendrer des temps de simulation et des temps de traitement informatique qui seraient rédibitoires.

3) DECOUPLAGE PAR LES CONSTANTES DE TEMPS

Des boucles ayant des constantes de temps très différentes peuvent être simulées avec des pas de calcul différents. Ceci peut être observé lorsque des phénomènes électriques, mécaniques et thermiques coexistent.

Ce type d'idée a été validé par différents travaux de recherche [DOE90] [NOR92]. Comme ce n'est pas une de nos préoccupations majeures, nous ne l'avons pas mis en oeuvre dans notre outil de simulation. Cependant, comme le générateur peut traiter des boucles séparément, il dispose des éléments suffisants pour être couplé avec un outil simulant plusieurs boucles avec des pas de calcul différents.

Les considérations que nous avons développées dans ce chapitre, nous ont conduit à spécifier les caractéristiques d'un outil pour la simulation des AMCC, dont nous avons réalisé le prototype. Nous allons présenter celui-ci dans le chapitre suivant.

Chapitre III

Présentation de notre outil de simulation

CHAPITRE III

PRESENTATION DE NOTRE OUTIL DE SIMULATION

Dans ce chapitre, nous présentons l'outil de simulation que nous avons élaboré à partir de la démarche présentée au chapitre II. Celui-ci se compose de trois modules principaux :

- un générateur d'équations et de programmes que nous avons entièrement élaboré, et qui est écrit en langage de calcul formel : Macsyma,
- un moteur de résolution d'équations différentielles qui existait déjà au sein du LEG et qui est utilisé dans le logiciel de simulation MC2 (Machine - Convertisseur - Commande),
- des interfaces de pré et post processeur qui existent également au laboratoire, et qui sont utilisées dans MC2.

Les équations d'état qui sont réorganisées et réduites par notre générateur, restent à la disposition de l'ingénieur ou du chercheur, pour faire des analyses autres que la simulation. Cela est développé à la fin de ce chapitre.

A. ORGANISATION DE L'OUTIL DE SIMULATION

1) ORGANISATION D'UNE ETUDE D'AMCC

Pour réaliser une étude d'AMCC avec notre outil de simulation, l'utilisateur doit procéder en plusieurs étapes (figure III.1).

Il commence par analyser son application avec l'objectif d'en faire une description détaillée suivant la démarche présentée au chapitre II. Le générateur effectue ensuite le traitement automatique des équations fournies par cette description, pour aboutir à un programme de simulation.

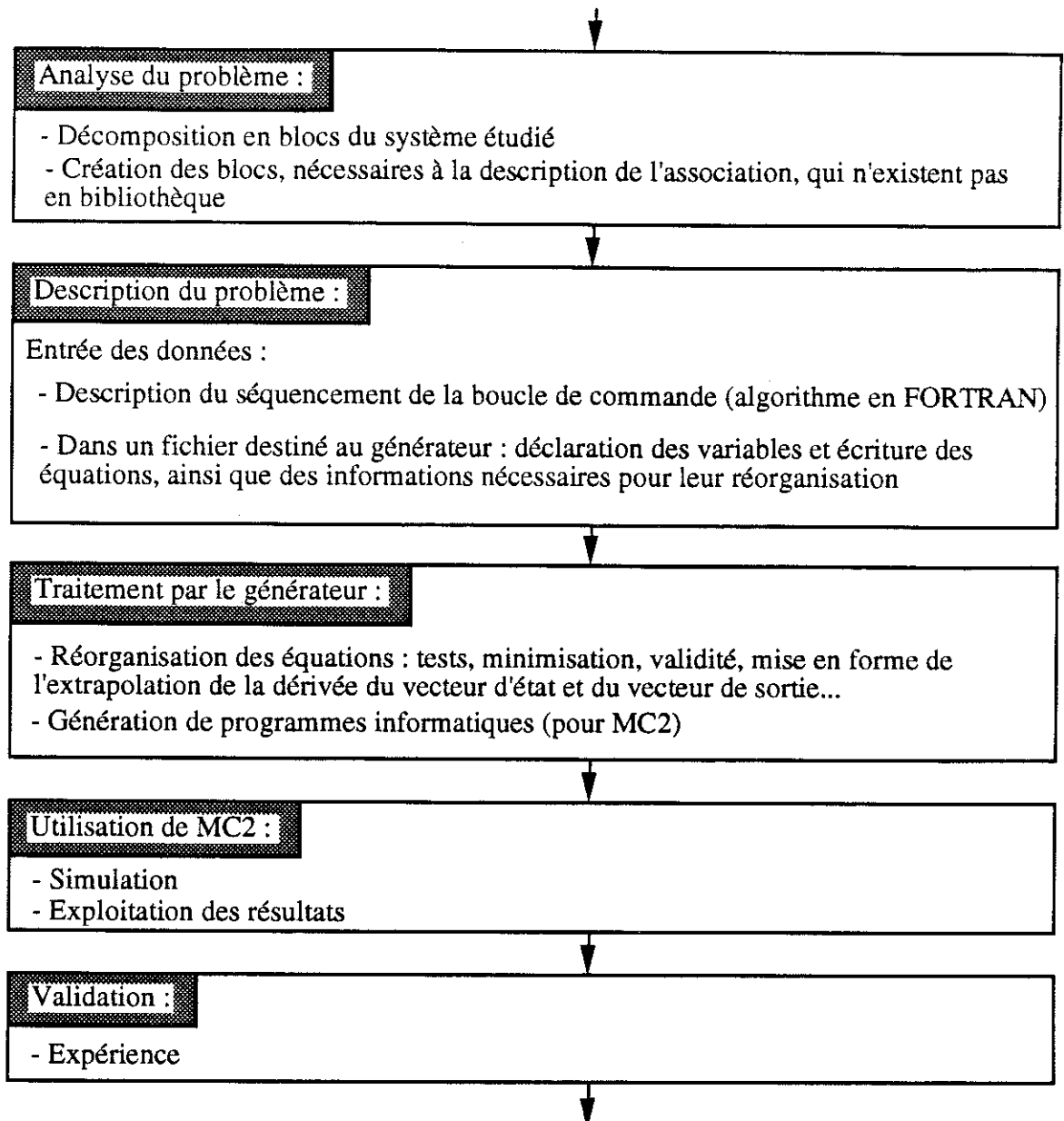


Figure III.1 : Organisation d'une étude d'AMCC

2) PARTIE GENERATEUR

Nous avons élaboré un générateur d'équations et de programmes de simulation pour passer automatiquement d'une description séparée des équations de puissance et de commande analogique, à une approche globale nécessaire à la résolution de ces équations (chapitre II).

Afin de s'assurer que les équations décrites par l'utilisateur sont cohérentes et compatibles, et qu'elles conduisent à des systèmes globaux physiquement réalistes, le générateur teste :

- l'association des différents blocs,

- la cohérence et la validité des équations des blocs,
- la singularité éventuelle des systèmes d'équations globaux obtenus,
- etc....

Les tests sont réalisés en utilisant des règles issues du savoir-faire que nous avons sur la modélisation et sur la simulation des AMCC.

Notre générateur réalise automatiquement tout le programme informatique correspondant aux parties puissance et commande analogique de l'association étudiée. En particulier, il réalise la préparation de la résolution différentielle des différents jeux d'équations d'état réorganisés, ainsi que la gestion de leur enchaînement (pour la simulation).

Notre générateur réalise également les programmes informatiques nécessaires aux interfaces utilisateur de l'outil de simulation final (voir paragraphe A.4)

3) PARTIE MOTEUR DE RESOLUTION

Cette partie, que l'on retrouve dans tout programme de simulation d'AMCC, doit offrir :

- des modules de résolution implicite d'équations différentielles (Runge Kutta 44, Euler...),
- des modules de gestion des événements importants tels que des franchissements de seuil de courant ou de tension, des tops d'horloge, des commandes d'interrupteur.

Afin de ne pas refaire un long travail de programmation, nous avons utilisé le logiciel de simulation d'AMCC qui était disponible au sein du LEG : MC2. Celui-ci répond aux deux critères présentés ci-dessus. Notons que tout logiciel équivalent aurait pu être employé.

Remarque : Dans MC2, la prise en compte des événements importants se fait par un mécanisme de prédiction [NOR92].

4) PARTIE INTERFACE UTILISATEUR

Pour pouvoir utiliser notre outil de simulation et pour en évaluer les performances, nous nous sommes servis, là aussi, des outils disponibles dans le laboratoire, et que l'on retrouve dans le logiciel MC2. Ceux-ci offrent :

- des moyens de vérification et de modification interactives des données,
- des moyens de sauvegarde des problèmes et des résultats,
- des moyens d'analyse des résultats : tracés de courbe.

Les interfaces sont conçues à partir de la bibliothèque MIC-CAO développée au laboratoire.

B . UTILISATION DE MACSYMA

1) LE CALCUL FORMEL

Le calcul formel, que l'on nomme aussi calcul symbolique, est un calcul analytique informatisé qui manipule des formules et des nombres exacts. Cela consiste, par exemple, à donner les solutions analytiques de l'équation du second degré :

$$ax^2 + bx + c = 0$$

Un logiciel de calcul formel peut donc s'employer pour faire des calculs analytiques, mais il est aussi assimilable à un langage de programmation de haut niveau [PIN] [MAC88]

2) INTERET DU LANGAGE DE CALCUL FORMEL

Nous présentons ici les différents éléments qui nous ont conduits à choisir l'emploi du calcul formel pour concevoir notre générateur.

a) Réalisation des fonctions de notre générateur

Contrairement à un langage de programmation classique (C++, Pascal, etc...), le calcul formel permet de réaliser efficacement, avec une programmation claire et rapide, un certain nombre de tâches que doit remplir notre générateur [GER92].

En particulier, avec un tel langage, nous pouvons :

- accéder aux informations contenues dans les équations,
- créer des procédures pour réduire la dimension des systèmes d'équations à résoudre (par l'utilisation des combinaisons linéaires entre certaines grandeurs),
- réorganiser formellement les équations sous les forme [1] ou [2] plutôt que sous la forme [3] (cf chapitre II),

- réaliser les tests de cohérence et de validité sur les équations. En particulier, nous pouvons gérer rigoureusement la réorganisation des équations pour les configurations de conduction discontinue (cf paragraphe III.C.4.c),
- générer uniquement les systèmes globaux d'équations représentant une configuration réelle de l'AMCC étudiée,
- etc...

b) Description des équations

Contrairement à la plupart des outils de simulation numérique d'AMCC, avec notre générateur, l'utilisateur fournit ses équations dans un formalisme naturel et sans aucun a priori sur la façon dont elles sont agencées et résolues.

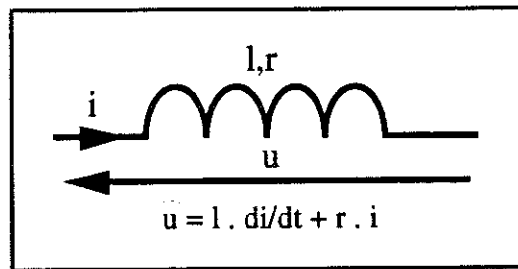


Figure III.2 : Equation d'une inductance

A titre d'exemple, l'équation de l'inductance représentée sur la figure III.2 s'écrit simplement :

$$l \cdot \frac{di}{dt} + r \cdot i = u ,$$

mais, toute autre écriture de cette équation est aussi acceptée par notre générateur :

$$l \cdot \frac{di}{dt} = u - r \cdot i \quad \text{ou} \quad \frac{u}{l} - \frac{r \cdot i}{l} = \frac{di}{dt}, \text{ etc..}$$

Une telle convivialité n'a pu être obtenue que par l'emploi du calcul formel dans notre générateur.

Remarque : Pour que notre générateur réorganise et simplifie les équations et pour qu'il réalise des tests sur celles-ci, nous avons souhaité introduire une part d'expertise issue de la modélisation et de la simulation des AMCC. Pour cela, nous avons utilisé les possibilités de

programmation du langage Macsyma : règles du type "si condition alors action" et programmation de fonctions.

Maintenant, nous allons présenter le langage de calcul formel que nous avons choisi.

3) MACSYMA

Parmi les langages de calcul formel commercialisés, nous avons choisi celui qui semblait le mieux adapté à la démarche que nous avons choisie : Macsyma [MAC88].

Macsyma est un outil interactif et un langage de programmation de haut niveau, permettant de résoudre un large spectre de problèmes mathématiques. Il offre de nombreuses possibilités pour les manipulations numériques et formelles (analytiques) concernant : l'algèbre, les calculs et l'analyse, les tracés graphiques 2D et 3D. Il offre aussi des possibilités d'interfaçage avec des processus mathématiques, et un environnement de programmation (cf annexe II).

La version 13 que nous utilisons, fonctionne sur PC 80-386, sous MS-DOS3, et dans l'environnement de WINDOWS3. Notons que Macsyma est disponible sous d'autres systèmes d'exploitation (Unix..).

C. LE GENERATEUR

1) STRUCTURE GENERALE

L'organigramme de la figure III.3 donne une idée générale du travail effectué par le générateur par rapport à celui qui est à la charge de l'utilisateur.

2) ENTREE DES DONNEES

La description d'une AMCC (respectivement toute modification de son modèle) impose la création (respectivement la modification) :

- du fichier d'entrée des données pour le générateur, en ce qui concerne la puissance et la commande analogique,
- de l'algorithme du séquençement de la commande et de l'appel des sous-programmes correspondant aux blocs TS_n, en ce qui concerne la commande numérique.

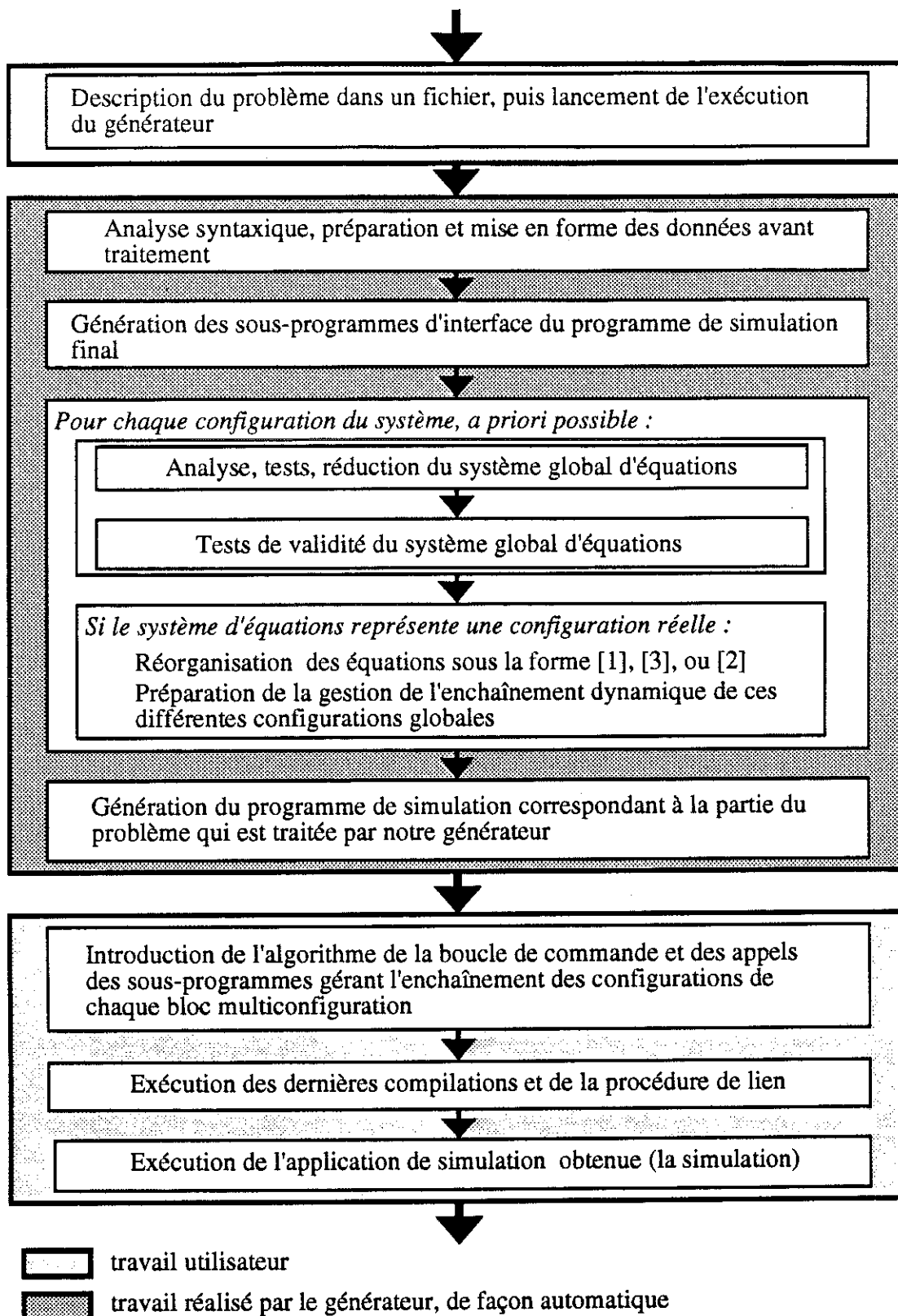


Figure III.3 : Emploi et structure du générateur

Par le jeu des symétries et des permutations d'indices, nous avons pu optimiser l'entrée des données, mais aussi limiter le nombre de systèmes d'équations à réorganiser.

3) REORGANISATION DES EQUATIONS

Pour chaque configuration du système global traité, le processus de réorganisation des équations que nous avons mis en oeuvre dans notre générateur, se fait en plusieurs étapes que nous présentons maintenant.

a) Tri et réduction des équations

Pour permettre les tests sur la cohérence des équations et afin de réduire chaque équation d'état globale, notre générateur trie celles-ci en différents groupes ou sous-groupes.

Il regroupe, en particulier, les équations du type $\sum_{i=1}^n a_i \cdot V_i = 0$ dont les a_i sont des

coefficients algébriques sans aucune représentation physique (valant le plus souvent ± 1).

Ensuite, ces combinaisons linéaires sont utilisées pour réduire la dimension du système d'équations global. Les tests de validité (paragraphe III.C.4) sont réalisés pendant ces opérations.

Si le système d'équations obtenu représente une configuration réelle de l'AMCC complète, il est simplifié et réorganisé sous une des trois formes présentées au paragraphe II.C.3.d.

b) Simplifications

Afin de simplifier les expressions obtenues pour les formes [1] et [2], nous avons élaboré des fonctions de simplification.

En effet, si les méthodes de calcul formel que nous avons employées pour obtenir ces formes donnent des résultats analytiques exacts, ces derniers peuvent être d'une taille considérable (plusieurs centaines de lignes...). Une des difficultés principales est de simplifier ces expressions, pour en réduire la dimension.

Les fonctions que nous avons élaborées dans ce but, sont réalisées en utilisant les symétries et la connaissance que nous possédons sur la modélisation des AMCC. En particulier, cela concerne des simplifications trigonométriques issues des modèles de Park (cf annexe V).

c) Obtention des formes de réorganisation

La forme [1] est obtenue par une méthode de résolution de système linéaire : Gauss formel. Pour les AMCC avec des équations de taille imposante, Gauss formel prend beaucoup de temps. Pour accélérer ce processus de réorganisation, nous avons essayé d'utiliser une méthode matricielle. Cette dernière permet d'utiliser des sous-expressions plus simples et plus courtes que Gauss, car le déterminant du système d'équations est traité séparément. Cependant, cette méthode donne des temps de réorganisation rédibitoires pour nos problèmes, donc nous l'avons rejetée.

La forme [2] est obtenue par une triangularisation formelle de Gauss.

La forme [3] correspond essentiellement à une analyse syntaxique des formules et à leur mise en forme pour une résolution numérique de systèmes linéaires.

4) GENERATION AUTOMATIQUE DE CODE INFORMATIQUE

Les résultats issus de cette réorganisation sont ensuite utilisés par notre générateur pour créer automatiquement un programme, en langage Fortran, permettant d'obtenir $\dot{X}_1(t)$, $Y_1(t)$ et $Y_2(t)$ pour chaque configuration du système (avec les notations du chapitre II.C.3.c.).

Pour les formes [1] et [2], cela correspond à une résolution séquentielle d'équations, alors que pour la forme [3], cela correspond d'une part à une résolution numérique par une méthode de type Gauss pour obtenir $\dot{X}_1(t)$ et $Y_1(t)$, et d'autre part, à une résolution séquentielle d'équations pour $Y_2(t)$.

Pour résoudre des problèmes de norme et de compilation Fortran, nous avons dû élaborer une procédure spécifique pour la programmation de calculs algébriques sur des expressions de taille "hors norme", c'est à dire pour toutes celles qui excèdent onze lignes dans le format Fortran.

Afin d'assurer et de démontrer la portabilité de notre générateur par rapport à un autre moteur de résolution que celui de MC2, nous avons réalisé et testé la possibilité de fournir des équations réorganisées en langage C.

5) TESTS SUR LES EQUATIONS

Nous avons élaboré ces tests essentiellement pour gérer des problèmes dûs à l'assemblage des blocs et à l'écriture des modèles.

Dans la phase de tri des équations, nous testons si des associations interdites apparaissent. En particulier, cela concerne la connection d'une source de courant en série avec une inductance, ou d'une source de tension en parallèle avec une capacité (figure III.4).

Pour éviter toute redondance d'équations ou de variables, et pour s'assurer que les systèmes sont a priori solubles, nous avons prévu des tests de singularité des systèmes d'équations globaux.

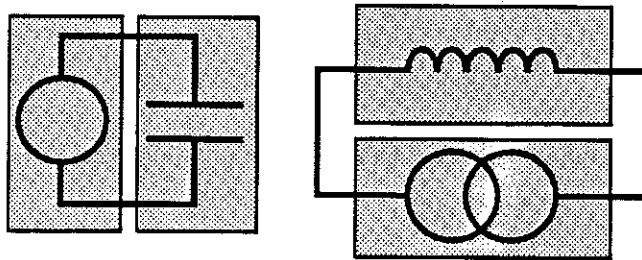


Figure III.4 : Connexions impossibles

Pour gérer les phases de conduction discontinue, nous avons dû élaborer des tests et des opérations spécifiques que nous présentons dans le paragraphe suivant.

Remarque : Pour éviter les risques éventuels de division par zéro lors du passage des équations réorganisées formellement à la simulation numérique, nous imposons une limitation sur l'écriture des modèles : "Les paramètres des équations doivent être non nuls". En effet, il est impossible de détecter ce type de division par le calcul formel, si on ne fait pas d'a priori sur la valeur des paramètres. Cette remarque concerne surtout la modélisation à topologie variable des convertisseurs statiques, où pour limiter le nombre de jeux d'équations, on peut être tenté de paramétrer ces derniers avec des coefficients pouvant s'annuler.

6) GESTION DE LA CONDUCTION DISCONTINUE

a) Le problème

Afin de ne pas générer des systèmes d'équations insolubles (voir § III.C.6.b) ou physiquement irréalistes (voir § III.C.6.c), les configurations de conduction discontinue des convertisseurs statiques modélisés à topologie variable, doivent être considérées avec prudence lors de la réorganisation des équations.

En effet, l'utilisateur ayant décrit ses blocs de puissance sans fournir aucune information sur l'influence d'un convertisseur sur un autre, notre générateur doit détecter automatiquement si deux convertisseurs sont nécessairement en conduction discontinue en même temps (voir § III.C.6.c).

Afin de clarifier ces aspects, nous les analysons à partir de l'exemple de la figure III.5. Nous nous intéressons plus particulièrement ici à l'ensemble redresseur - filtre - commutateur de courant. Cette structure est utilisée dans différentes AMCC (avec certaines variantes), telles que la machine asynchrone autopilotée alimentée en courant, par exemple.

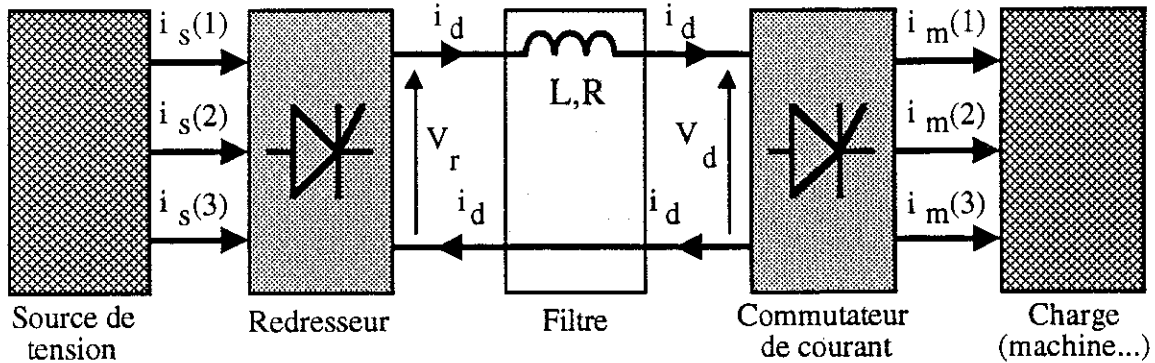


Figure III.5 : Gestion de la conduction discontinue lors de la réorganisation des équations

b) Traitement de la conduction discontinue simultanée dans plusieurs convertisseurs

Lorsque deux convertisseurs statiques, séparés par un filtre, entrent simultanément en conduction discontinue, il est possible que certains potentiels deviennent flottants, donc impossibles à calculer avec un modèle à topologie variable. Notre générateur détecte et traite alors ce problème automatiquement, comme nous allons le montrer autour de l'exemple de la figure III.5.

La phase de conduction discontinue globale est caractérisée par l'annulation des courants dans chaque convertisseur. Ainsi, dans notre exemple, si les courants i_d ou $\{i_s(1), i_s(2) \text{ et } i_s(3)\}$ ou $\{i_m(1), i_m(2), i_m(3)\}$ s'annulent, on montre aisément que les deux convertisseurs statiques entrent en même temps en conduction discontinue. Ceci introduit une redondance dans la résolution des équations de l'ensemble de la chaîne de puissance.

En effet, en conduction discontinue, les équations sont :

$$\text{- pour le redresseur : } i_s(1) = 0 \quad (1)$$

$$i_s(2) = 0 \quad (2)$$

$$i_s(3) = 0 \quad (3)$$

$$i_d = 0 \quad (4)$$

$$\text{- pour le filtre : } L \cdot \frac{di_d}{dt} + R \cdot i_d = V_r - V_d \quad (5) \quad (i)$$

- pour le commutateur : $i_m(1) = 0$ (6)

$$i_m(2) = 0 \quad (7)$$

$$i_m(3) = 0 \quad (8)$$

$$i_d = 0 \quad (9)$$

Ainsi, l'équation $i_d = 0$ apparaît deux fois, par contre, une seule relation existe pour calculer les 2 grandeurs V_r et V_d , qui sont des potentiels flottants (c'est-à-dire inconnus).

Dans un tel cas, il est nécessaire d'éliminer une des deux équations $i_d = 0$ et une des variables V_r ou V_d du système global (i). Ici, comme $i_d = 0$ (c'est à dire $i_d = 0$ et $di_d/dt = 0$), l'équation (5) devient $V_r = V_d$. Un système non singulier peut être obtenu. Afin de permettre la résolution des équations au cours de la simulation, nous avons choisi de faire l'hypothèse que V_d et V_r sont constants pendant la phase de conduction discontinue et égaux à la dernière valeur qu'ils avaient juste avant (on néglige les pertes dans le convertisseur). Comme les variables V_r et V_d ne sont pas des variables d'état, elles peuvent subir des discontinuités. Ainsi, dès la remise en conduction continue des convertisseurs, elles reprennent une valeur correcte.

c) Suppression des configurations impossibles

Il est important de limiter le temps mis par le générateur pour réorganiser les équations et du même coup, de limiter la taille du programme de simulation final. Pour ne pas engendrer de jeux d'équations globaux qui représenteraient des configurations concrètement impossibles, notre générateur veille à ne conserver que les jeux d'équations utiles, c'est-à-dire ceux qui correspondent à des configurations réelles de l'AMCC étudiée.

Pour illustrer ces propos, reprenons l'exemple ci-dessus. Pour le redresseur, on peut prendre un modèle avec :

- une configuration pour la conduction discontinue (cr 1)
- une configuration pour la conduction normale (en indiciant les tensions de la source) (cr 2)

et pour le commutateur :

- une configuration pour la conduction discontinue (cc 1)
- six configurations pour les phases de conduction normale (cc 2 à cc 7)
- six configurations pour les phases d'empiètement d'anode (cc 8 à cc 13).

Par des tests et des traitements d'équations, notre générateur détecte que ces deux convertisseurs ne peuvent pas être séparément en conduction discontinue. Donc, il ne réorganise que $13 = (1 + 1 \times 12)$ jeux d'équations globaux au lieu de $26 = (2 \times 13)$ (résultat qui serait donné par des considérations purement algébriques). Les 13 configurations correspondent à cr 1 - cc 1 et à cr 2 - cci ($2 \leq i \leq 13$).

Ainsi, ce sont uniquement des jeux d'équations globaux physiquement possibles qui sont construits.

d) Information à fournir pour chaque convertisseur

Notre générateur gère les aspects précédents (§ III.C.6.b et § III.C.6.c) automatiquement, grâce à une information que l'utilisateur doit fournir lors de la description de chaque convertisseur. Cette indication permet de caractériser la phase de conduction discontinue du convertisseur par les courants qui s'annulent. Par exemple, le redresseur de la figure III.5 est en conduction discontinue si et seulement si $i_d = 0$ ou $\{i_s(1) = 0, i_s(2) = 0 \text{ et } i_s(3) = 0\}$. Nous rappelons qu'ici, aucun a priori n'est fait sur l'influence de la conduction discontinue d'un convertisseur sur un autre. Notre générateur sait le déduire automatiquement.

7) TESTS SUR L'ENTREE DES DONNEES (erreurs de description)

Certaines erreurs d'écriture des données sont détectées par notre générateur. Cette détection se fait par des tests de cohérence des équations et par des tests sur la description des données de l'AMCC étudiée.

Ainsi, notre générateur détecte s'il manque une information pour caractériser une symétrie, une permutation d'indices dans un convertisseur, ou si les données ne rentrent pas dans le formalisme de description, notamment au niveau de la nomenclature des grandeurs (cf annexe III).

Les tests sur les équations permettent de soulever certaines incohérences dues à une mauvaise entrée des équations (cf paragraphe III.C.5) qui risqueraient d'entraîner des singularités.

Remarques : Les erreurs typographiques non détectées par notre générateur ainsi que des mauvaises initialisations de grandeurs ne pourront être décelées qu'au niveau de la compilation des fichiers informatiques générés, ou lors de l'exécution du programme de simulation final.

La figure III.6 fait la synthèse du travail réalisé par l'utilisateur et par notre générateur, pour obtenir une simulation d'AMCC.

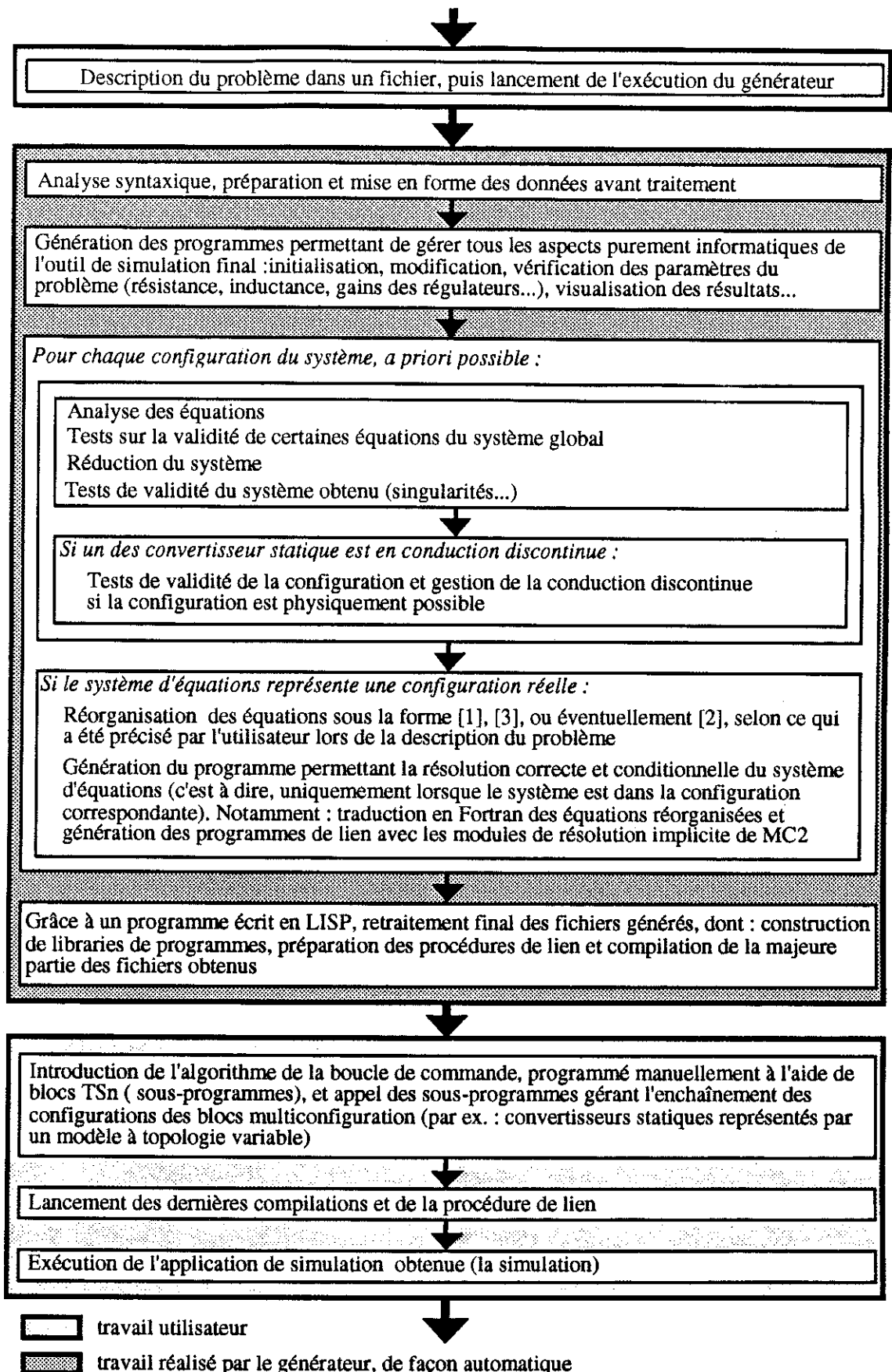


Figure III.6 : Emploi et structure du générateur

D. OUVERTURES OFFERTES PAR L'EMPLOI DU CALCUL FORMEL

1) OUVERTURE À PARTIR DES EQUATIONS REORGANISEES PAR NOTRE GENERATEUR

Contrairement aux formes [2] et [3] qui sont surtout exploitables pour la simulation, les équations mises sous la forme [1] peuvent être utilisées pour d'autres analyses que la simulation.

Rappelons que cette forme est une réduction de l'équation d'état globale :

$$f(\dot{X}(t), X(t), Y(t), u(t), \theta(t), t) = 0$$

pour obtenir un système d'équations indépendantes, qui s'écrit sous la forme matricielle :

$$\begin{aligned} \dot{X}_1 &= A(\theta).X_1 + B(\theta).u \\ Y_1 &= C(\theta).X_1 + D(\theta).u \end{aligned} \quad (1)$$

pour chaque configuration de l'AMCC étudiée.

Nous allons maintenant présenter certaines possibilités d'analyse offertes par ces équations réorganisées.

a) Apport pour la simulation

A partir de la forme [1], on peut, dans certains cas, ajuster le pas de calcul pour la simulation. En effet, par l'utilisation de Macsyma, il est possible de connaître les expressions formelles des valeurs propres de la matrice $A(\theta)$ si sa dimension est inférieure ou égale à quatre et les valeurs propres numériques au delà (il n'existe pas de solution analytique générale si la dimension est supérieure à quatre).

Ainsi, connaissant ces valeurs propres, nous pouvons ajuster le pas de calcul si le système étudié n'a pas de constantes de temps plus contraignantes que celles données par $A(\theta)$. Par exemple, c'est la commande qui impose le pas de calcul lorsque sa période d'échantillonnage est plus faible que les constantes de temps électriques (cas d'une commande de convertisseur en modulation de largeur d'impulsion).

b) Etude de l'influence de paramètres

Les équations formelles peuvent aussi servir à étudier l'influence de certains paramètres sur le comportement global de l'AMCC.

Par exemple, elles peuvent permettre d'ajuster la dynamique de l'AMCC, en modifiant les valeurs des paramètres intervenant dans les constantes de temps formelles fournies par $A(\theta)$. Les études peuvent se faire par la simulation, mais aussi la dérivation formelle offerte par Macsyma.

c) Etudes pour l'automatique

Si le système (1) est linéaire, c'est à dire de la forme

$$\dot{X}_1 = A.X_1 + B.u$$

$$Y_1 = C.X_1 + D.u$$

toutes les règles de commandabilité et d'observabilité issues de la théorie de représentation des systèmes d'état sont directement applicables. Pour cela, on utilise les matrices A, B et C [GIL75] [JAU89] [FOU87].

Là encore, les manipulations matricielles analytiques sont grandement facilitées par l'emploi du calcul formel.

De même, pour l'étude de la synthèse de commande et de régulation, il peut être une excellente base pour chaque configuration de l'AMCC étudiée [MID76] [CAT92].

Ceci concerne surtout les convertisseurs continu-continu (Buck, Boost, ...), mais aussi, tous les convertisseurs commandés en modulation de largeur d'impulsion (MLI) et utilisés pour alimenter des machines électriques.

Remarque : Ce type de calcul peut être automatisé avec un langage de calcul formel.

2) UTILISATION DU CALCUL FORMEL POUR DES CALCULS COMPLEXES

a) Obtention de modèles d'ordre réduit pour la synthèse des régulations

Pour concevoir des commandes et pour faire la synthèse de régulateurs, il est souvent nécessaire d'utiliser des modèles d'ordre réduit (au sens de l'automatique). Ces derniers peuvent être obtenus par des méthodes analytiques de réduction de modèle, par exemple [JOU91] :

- en passant du modèle d'état à la fonction de transfert, puis en éliminant les modes rapides ou en utilisant une réduction de Markove et Pade,
- en fixant l'équation caractéristique du modèle d'ordre réduit.

Tous ces calculs peuvent être longs et complexes, mais ils sont automatisables par un langage tel que macsyma. Les méthodes peuvent être mises en oeuvre avec des outils d'analyse numérique. De même, les modèles d'ordre réduit peuvent aussi être obtenus par des identifications du modèle initial (d'ordre élevé) ou de l'expérience [JOU91]. Cependant, ces approches numériques n'offrent pas la généralité fournie par les modèles analytiques.

b) Obtention d'observateurs ou d'estimateurs d'état

Pour concevoir des estimateurs et des observateurs d'état, on doit réaliser des calculs analytiques qui sont facilités et accélérés par l'utilisation d'un langage de calcul formel. Ces estimateurs et observateurs sont utilisés pour calculer (c'est-à-dire "capter") des grandeurs qui sont nécessaires à la commande, mais qui sont difficiles d'accès par des capteurs (composants). Par exemple, des observateurs sont utilisés dans la commande vectorielle pour estimer les flux rotoriques ϕ_{rd} et ϕ_{rq} de la machine, à partir des courants statoriques i_d et i_q , ou pour estimer les pulsations des courants rotoriques et statoriques.

c) Manipulation analytique de modèles

Le calcul ou l'inversion matricielle d'un modèle de machine peut être grandement facilité par le calcul formel. L'annexe V donne une idée de la complexité de ces modèles, dans le cas de la machine synchrone.

3) BILAN SUR LE CALCUL FORMEL

L'emploi du calcul formel est donc très intéressant pour limiter le temps de développement des calculs analytiques et pour augmenter la fiabilité de ceux-ci. Il apparaît comme un outil d'analyse très performant et très fiable. Les premiers logiciels de calcul formel ont été développés au début des années 1970, et sont maintenant utilisés dans de nombreux domaines scientifiques [ALG91]. L'annexe II résume les principales possibilités de Macsyma.

E. QUELQUES CHIFFRES SUR NOTRE GENERATEUR

Notre générateur est constitué de fichiers de commandes en Macsyma, qui occupent 250 Koctets. Le langage Macsyma occupe 25 Moctets et fonctionne dans l'environnement de WINDOWS 3.

Nous avons obtenu un outil de simulation qui répond aux exigences de la démarche que nous proposons au chapitre II. Par l'emploi du calcul formel, nous avons pu optimiser la

réorganisation des équations (tests, simplifications, réductions), et nous avons pu étendre à d'autres études les possibilités de notre outil.

En effet, notre générateur génère des programmes de simulation, mais aussi des jeux d'équations que l'ingénieur, ou le chercheur, peut utiliser pour des analyses.

Nous allons maintenant présenter certaines simulations que nous avons obtenues avec notre outil, et que nous avons validées expérimentalement.

Chapitre IV

Application et résultats

CHAPITRE IV

APPLICATION ET RESULTATS

La démarche de simulation numérique d'AMCC, présentée dans les chapitres II et III, a été testée pour la réorganisation des équations et pour les tests faits par notre générateur, sur plusieurs applications traitées au sein du LEG.

La démarche complète de simulation a été validée pour les deux exemples traités dans ce chapitre. Ceux-ci ont été choisis, d'une part, parce que pour chacun d'eux nous disposions d'un banc d'essai au laboratoire, et d'autre part, parce qu'ils permettent de bien mettre en évidence les problèmes que l'on peut rencontrer dans les outils de simulation classiques. Ces deux exemples sont traités complètement et validés expérimentalement.

A. MACHINE A COURANT CONTINU ALIMENTEE PAR UN HACHEUR QUATRE QUADRANTS COMMANDE EN MODULATION DE LARGEUR D'IMPULSION

La figure IV.1 présente les détails de la partie puissance de l'AMCC considérée.

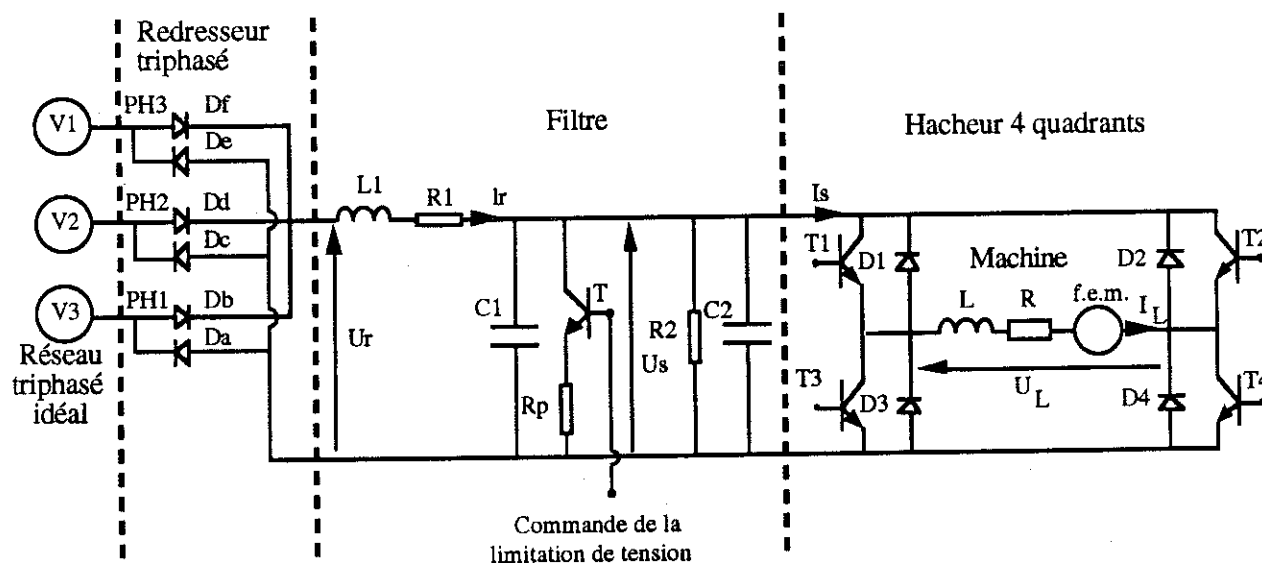


Figure IV.1 : Partie puissance de l'association étudiée

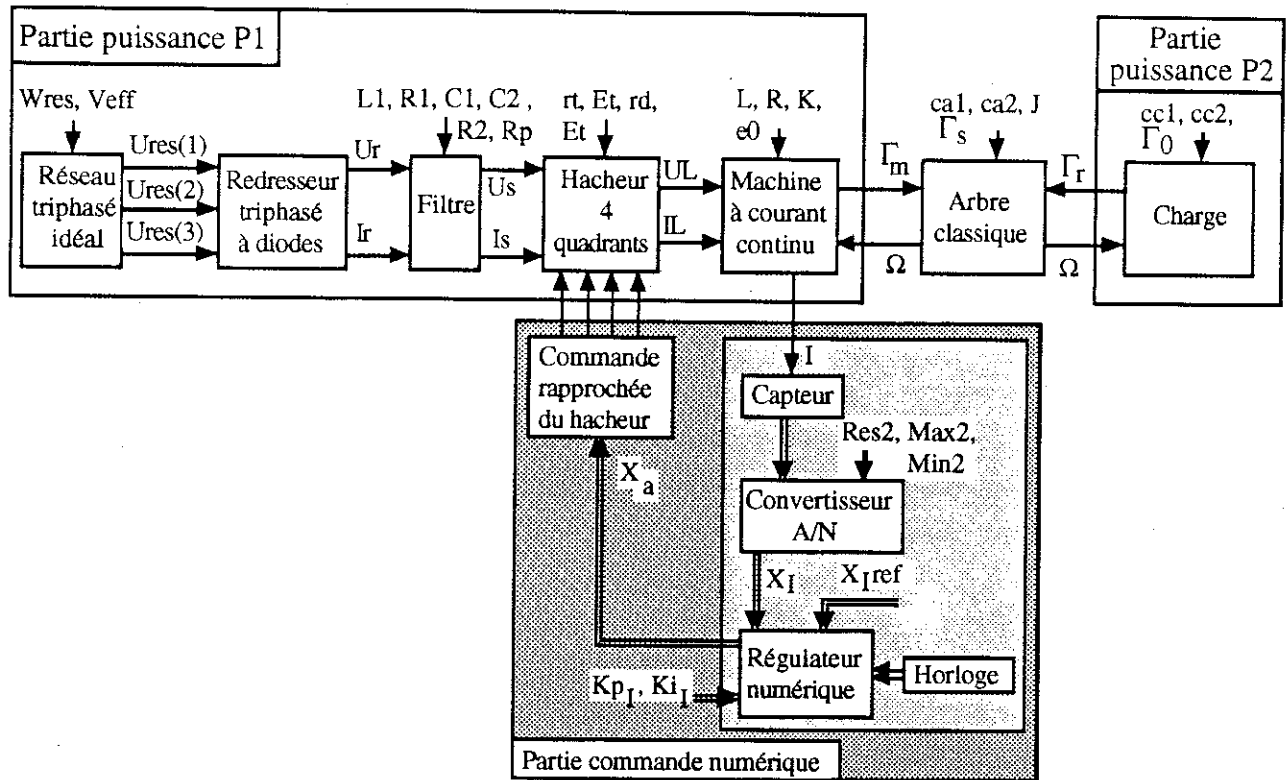


Figure IV.2 : Décomposition en blocs de l'association étudiée

La figure IV.2 présente une décomposition en blocs de l'association complète. Cet exemple permet sur un cas simple de :

- vérifier aisément et "à la main" le travail réalisé automatiquement par le générateur : réorganisation des équations et réalisation de programmes informatiques,
- vérifier la modélisation à topologie variable des convertisseurs statiques : génération et gestion de l'enchaînement des configurations,
- vérifier la possibilité de modéliser la commande numérique en prenant en compte la période d'échantillonnage, les temps d'exécution des fonctions et la numérisation (quantification des grandeurs),
- valider expérimentalement la simulation [CAT92].

1) DESCRIPTION DU PROBLEME

Comme le montre la figure IV.2, cette association comporte deux parties de puissance : P_1 et P_2 , qui sont découplées par l'arbre (cf chapitre II.D).

La **partie P₁** comporte :

- un réseau triphasé supposé idéal,
- un redresseur à diodes,
- un filtre de tension avec une limitation de surtension qui sert aussi de circuit dissipatif (pour le freinage),
- un hacheur quatre quadrants commandé en modulation de largeur d'impulsion (MLI),
- une machine à courant continu, représentée par son modèle le plus simple :

$$L \cdot \frac{dI_L}{dt} + r \cdot I_L + e + e_b = U_L$$

où U_L : tension aux bornes de la machine,

I_L : courant dans la machine,

$e = k \cdot \Omega$,

e_b : chute de tension au niveau des balais de la machine,

$\Gamma_m = k \cdot I_L$: couple moteur.

La **partie P₂** est une charge mécanique dont le couple est donné par l'équation

$$\Gamma_{ch} = \Gamma_0 + cc1 \cdot \Omega + cc2 \cdot \Omega^2$$

où Γ_0 , $cc1$, $cc2$ sont des constantes,

Ω : vitesse angulaire de l'arbre,

θ : position angulaire de l'arbre.

L'**arbre** est défini par l'équation :

$$J \cdot d\Omega/dt = \Gamma_m - \Gamma_{ch} - \Gamma_s - ca1 \cdot \Omega - ca2 \cdot \Omega^2$$

où, J : moment d'inertie,

Γ_s : couple de frottement sec,

$ca1$, $ca2$ sont des constantes qui caractérisent des frottements visqueux, par exemple.

La **commande** est entièrement numérique. Dans cet exemple, nous n'avons implanté que la boucle fermée de régulation en courant. Si une boucle de vitesse avait été implantée, la commande aurait la structure décrite sur la figure IV.3. Cette boucle de vitesse serait construite de la même façon que la boucle de courant. Son implantation dans l'outil de simulation serait donc simple.

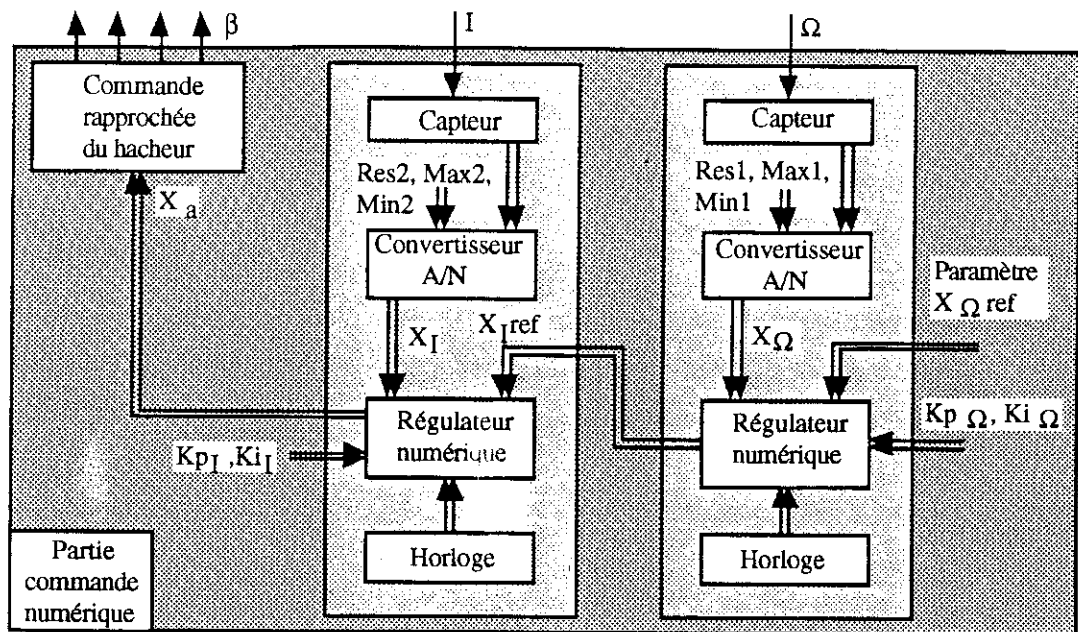


Figure IV.3 : Structure de la commande en vitesse et en courant

2) SPECIFICITES DU PROBLEME

a) Causalité

Si nous ne considérons pas la phase où les deux convertisseurs (redresseur et hacheur) sont systématiquement en conduction discontinue, la causalité (cf chapitre I) entre les différents blocs est identique pour toutes les configurations de l'association. En effet, aucun partage des dérivées de variable d'état n'existe entre les blocs définis sur la figure IV.1. Ce type d'application peut donc être traité par les outils de simulation classiques.

Pour ces configurations (sans la conduction discontinue), nous pourrions résoudre séquentiellement les blocs de la partie P1, en commençant par la source et en terminant par la machine. Cependant, ceci supposerait de décrire les blocs utilisés pour P1, avec un a priori sur la manière dont ils sont résolus (sur le séquençement de la résolution); ces blocs, liés à une résolution, seraient donc très spécifiques, et ainsi peu réutilisables.

b) Modélisation à topologie variable

Sur un exemple simple, cette application montre les différents aspects de la mise en oeuvre d'un modèle à topologie variable pour les convertisseurs statiques qui sont :

- la mise en oeuvre des tests de commutation des interrupteurs.(on peut employer la représentation par réseaux de Pétri [HAU] [MAN88]).
- le choix du modèle des interrupteurs à l'état passant.

Remarque : à l'état bloqué, en topologie variable, les interrupteurs sont toujours représentés par un circuit ouvert.

Nous avons obtenu deux configurations pour le redresseur à diodes (voir annexe III.1). A l'état passant, nous avons modélisé les diodes par des courts-circuits.

Pour le hacheur, nous avons réalisé un modèle plus complet. Nous avons modélisé les interrupteurs par une faible résistance et par une chute de tension constante à l'état passant. De plus, nous avons différencié la valeur de ces paramètres pour les diodes et les transistors. En tenant compte des symétries du hacheur, nous avons défini uniquement quatre configurations en fonctionnement normal.

c) Modélisation à topologie fixe

Pour le filtre, deux états doivent être modélisés. En utilisant une résistance linéaire de valeur $R_2 // R_p$ ou R_2 (figure IV.1), selon l'état de conduction du transistor T, un seul jeu d'équation suffit pour représenter le fonctionnement de ce filtre.

Des considérations identiques permettent de modéliser la chute de tension e_b aux balais de la machine par $e_b = (+e_0)$ ou $(- e_0)$ selon le signe du courant I_L dans la machine.

d) La commande numérique

Nous avons simulé les temps réels d'exécution des fonctions de la commande numérique, ainsi que leurs discrétisations numériques et temporelles. Ceci a été grandement facilité par la structure que nous avons donnée aux blocs TS_n (cf annexe VI). Nous avons mis en oeuvre le séquençement, à l'aide d'un algorithme qui organise l'appel des sous-programmes représentant les blocs TS_n .

3) RESULTATS

Forme de réorganisation des équations d'état	Durée d'exécution du générateur sur PC 386 - 33MHz	Taille du fichier source généré pour les équations d'état de la partie puissance P1 (.FOR)	Taille finale du programme généré (.EXE)	Durée de la simulation sur PC 386 - 33MHz, pour un temps réel simulé de 0 à 0,1s et un pas par défaut de 0,00002s
Forme (1)	22,20 s	10628 Octets	458930 Octets	55,63 s
Forme(2) avec simplifications supplémentaires spécifiques aux équations de machine	32,30 s	10570 Octets	458406 Octets	56,72 s
Forme (2) avec simplifications supplémentaires spécifiques aux équations de machines et optimisations pour grosses formules	45,52 s	10620 Octets	458948 Octets	57,97 s
Forme (3) un sous - programme par configuration global de l'association	26,22 s	16258 Octets	468836 Octets	3 mn 2,47s
Forme (3) toutes les configurations de l'association sont réunies dans un même programme	26,22 s	16258 Octets	465932 Octets	2 mn 39,44s

*Figure IV.4 : Résultats de simulation :
comparaison entre les différentes formes de réorganisation*

Par rapport à une programmation directe avec un outil de simulation classique d'AMCC, l'approche avec notre générateur nous a permis ici :

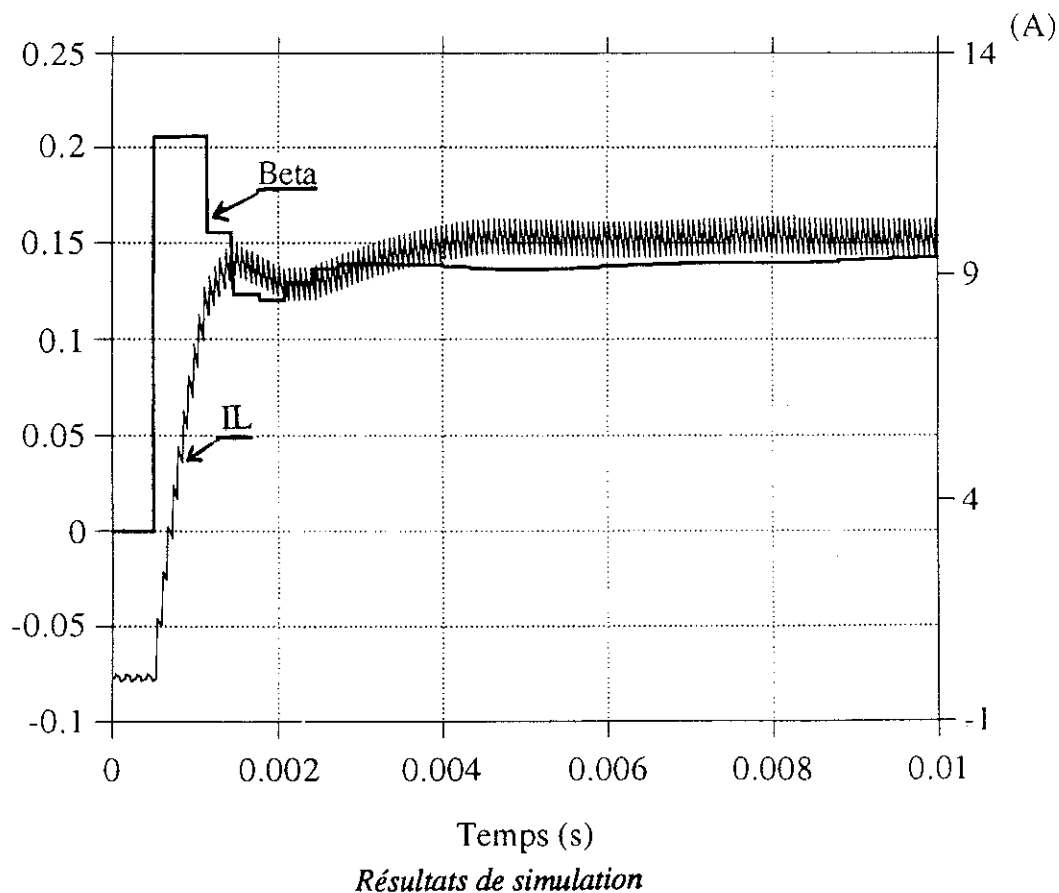
- de gagner en simplicité et en rapidité, en particulier, pour décrire les données de l'association et pour réaliser le programme de simulation,
- de ne pas nécessiter de connaissances sur la structure de l'outil utilisé pour résoudre dynamiquement les équations différentielles. Seul le format des données (norme

Fortran) et le séquençement de la boucle de commande numérique restent un a priori à connaître pour l'utilisateur,

- d'offrir l'avantage de décrire les équations dans un formalisme naturel, sans a priori sur la façon dont elles sont résolues,
- de vérifier l'aisance de "programmation" des blocs qui ont de nombreuses configurations avec des équations plus ou moins complexes au niveau du modèle des interrupteurs (pour les convertisseurs),
- de montrer la possibilité de modéliser finement la commande numérique.

Les trois formes de réorganisation des équations ont été testées. Nous avons imposé des options supplémentaires pour la mise en forme et l'obtention du programme final de simulation.

Le tableau IV.4 résume les résultats que nous avons obtenus sur un PC 80-386-33 Hz. Les résultats des simulations sont identiques pour les trois formes de réorganisation. Ils ont été validés par l'expérience (cf figure IV.5).



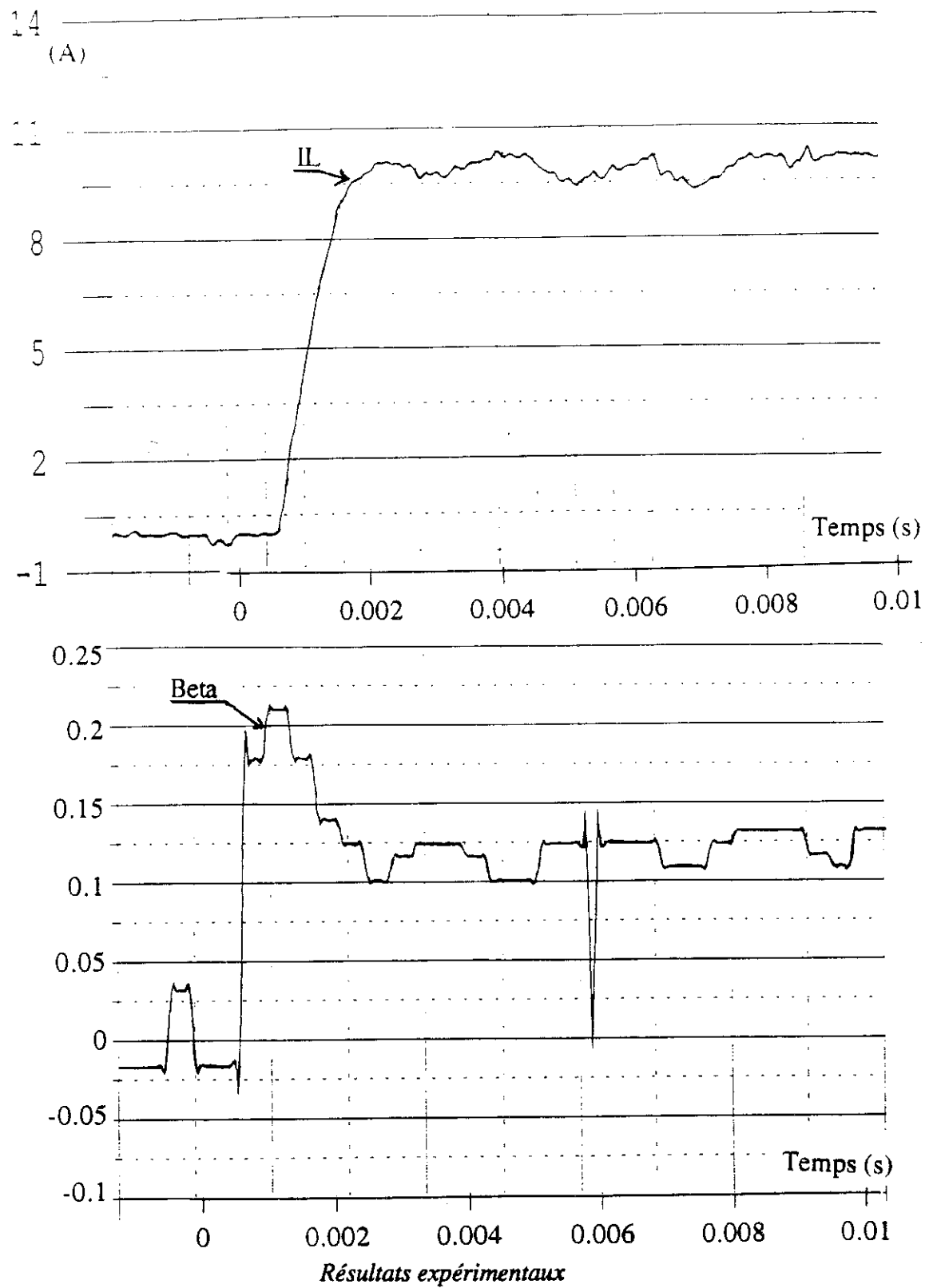


Figure IV.4 : Résultats de simulation : comparaison avec l'expérience

B. MACHINE SYNCHRONES AUTOPILOTEE ALIMENTEE EN COURANT

L'association étudiée est représentée sur la figure IV.6 et sa décomposition fonctionnelle en blocs sur la figure IV.7.

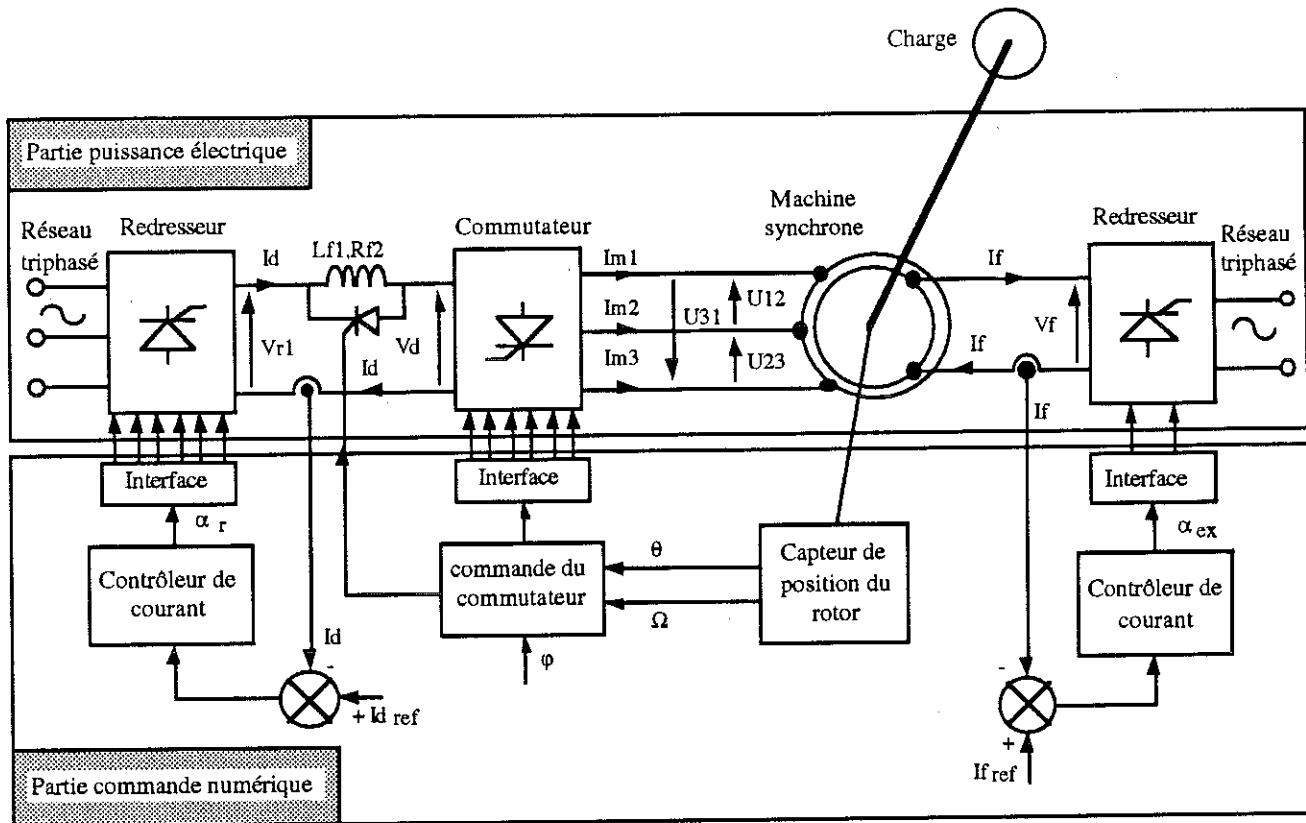


Figure IV.6 : Schéma de l'association étudiée

Nous avons choisi cette application car :

- elle permet de valider notre démarche sur un des systèmes les plus complexes, que l'on ait à étudier (voir les équations de la machine en annexe V),
- la causalité entre les différents blocs de puissance évolue selon les configurations de l'association : une résolution globale des équations de puissance est donc indispensable,
- cette association permet de valider le travail réalisé par notre générateur, pour traiter les problèmes des phases de conduction discontinue dans les convertisseurs (cf chapitre III.C.6) et pour gérer un nombre important de configurations,
- une validation expérimentale de la simulation a pu être réalisée [PAZ92].

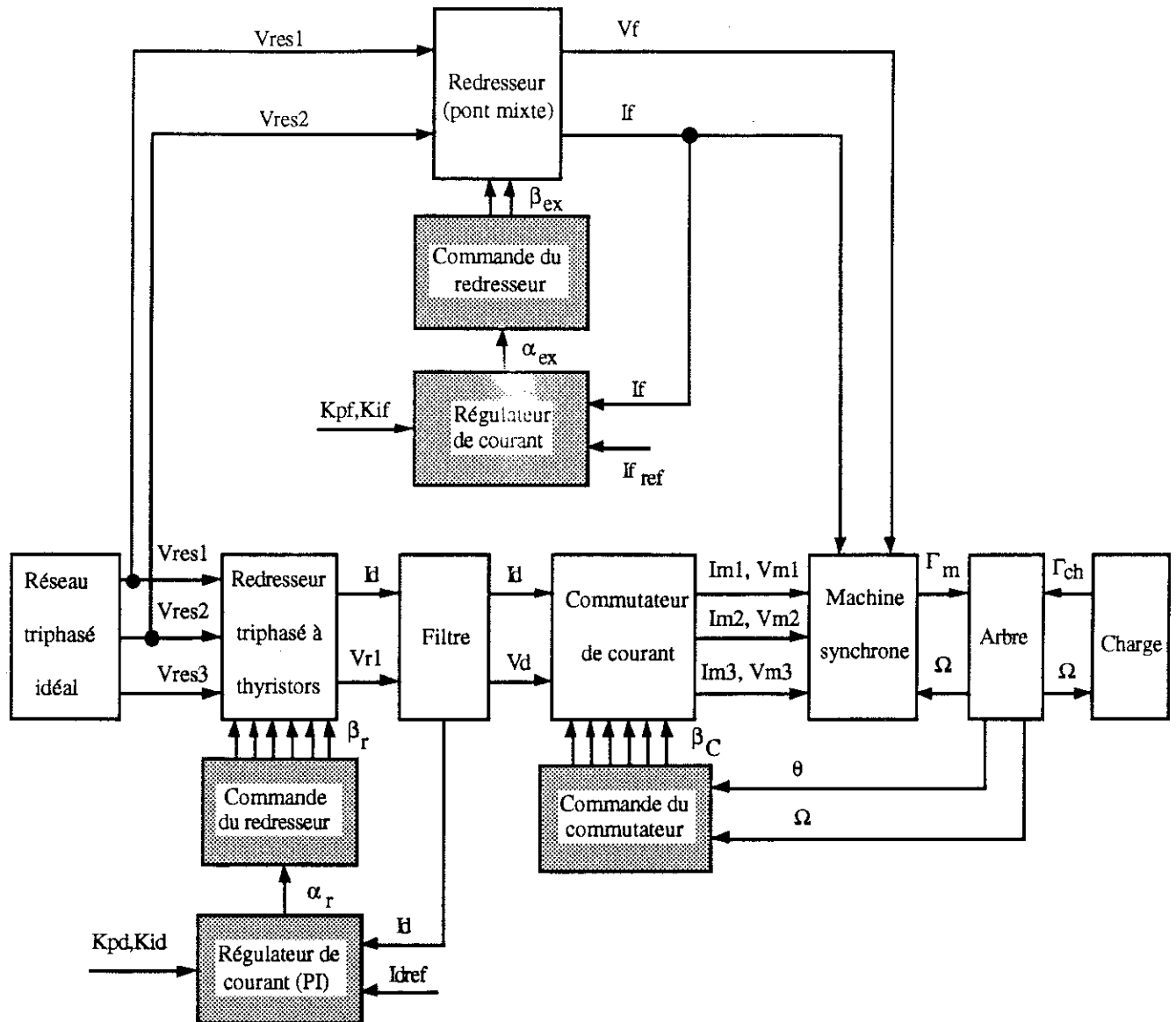


Figure IV.7 : Décomposition en blocs de l'association étudiée

1) DESCRIPTION DU PROBLEME

Reprenons la figure IV.6 pour décrire ce problème.

La machine synchrone comporte :

- Un inducteur alimenté à partir du réseau électrique (supposé idéal), par l'intermédiaire d'un pont redresseur mixte monophasé.

- Un induit alimenté à partir du réseau électrique, par l'intermédiaire d'un redresseur triphasé à thyristors, puis d'un filtre de courant, et enfin, d'un commutateur de courant (à thyristors). Ce dernier permet l'autopilotage de la machine. Il fonctionne en commutation naturelle, sauf à basse vitesse. Dans ce dernier cas, il faut forcer l'annulation du courant I_d dans la branche continue, d'où la présence du thyristor placé en parallèle avec l'inductance de lissage du courant I_d .
- Une cage d'amortisseur.

Le courant en ligne I_d (côté stator) et le courant d'excitation I_f (côté rotor) sont régulés.

Nous avons implanté deux modèles pour la partie {arbre + charge} :

- le premier impose une vitesse constante (fournie par l'utilisateur en début de simulation),
- le second est semblable à celui présenté au paragraphe IV.A.2 (un arbre rigide et une charge mécanique).

2) SPECIFICITES DU PROBLEME

a) La commande numérique

Pour la commande numérique, cette application est équivalente à celle présentée au paragraphe IV.A.

b) Nécessité d'une résolution globale

Une résolution globale des équations de l'ensemble des blocs : {filtre, commutateur, machine synchrone} est nécessaire. En effet, par l'intermédiaire du commutateur, le courant I_d est commun au filtre et à la machine, en particulier, sa dérivée apparaît dans ces deux blocs (cf chapitre I.B.3).

c) La complexité des équations de la machine

Comme le montre l'annexe V, les équations de la machine sont complexes et lourdes à manipuler analytiquement [PAZ92]. Cependant, elles présentent des symétries trigonométriques

qui sont prises en compte par notre générateur. De plus, ce dernier opère des simplifications qui permettent d'obtenir des équations de taille plus réduites (mais ceci reste relatif !).

d) Les équations des convertisseurs

Les convertisseurs sont représentés par des jeux d'équations qui lient linéairement certaines grandeurs entre elles. Par exemple, les relations décrites dans le commutateur sont des combinaisons linéaires entre ses courants amont (I_d) et aval (I_{mi} au stator de la machine), et entre ses tensions amont (V_d) et aval (U_{ij} au stator de la machine).

Grâce à de telles relations, notre générateur réduit le jeu d'équations de chaque configuration globale de l'association avant de le réorganiser sous la forme [1], [2] ou [3]. Cette réduction est très importante, car toute résolution de Gauss (formelle ou numérique) se fait en un nombre d'opérations qui est proportionnel au cube de la dimension du système d'équations à résoudre.

e) La conduction discontinue

Dans cette association, on sait que le redresseur triphasé et le commutateur de courant sont nécessairement en conduction discontinue en même temps. En conséquence, pour cette configuration, notre générateur gère automatiquement le problème des potentiels flottants V_{r1} et V_d , qui est exposé au paragraphe III.C.6.

De même, notre générateur ne génère que les configurations globales physiquement possibles.

Nous avons représenté les quatre blocs multiconfiguration de cette association par :

- . $n_1 = 2$ configurations pour le redresseur triphasé,
- . $n_2 = 2$ configurations pour le redresseur monophasé,
- . $n_3 = 13$ configurations pour le commutateur de courant,
- . $n_4 = 2$ configurations pour le filtre de courant situé entre le redresseur triphasé et le commutateur.

Notre générateur n'a alors généré que $n_2 \times n_4 \times [(1 \times 1) + (n_1 - 1)(n_3 - 1)] = 52$ configurations physiquement possibles (cf paragraphe III.C.6).

f) Gestion de l'enchaînement des configurations du commutateur

Rappelons que la gestion de l'enchaînement des configurations de **chaque** bloc, est décrite par l'utilisateur. A partir de ces descriptions séparées, pour chaque bloc, notre générateur réalise automatiquement une gestion de l'enchaînement des configurations globales de l'AMCC étudiée.

Dans cette application, le commutateur de courant est le bloc le plus complexe à décrire en raison du nombre élevé de tests à mettre en oeuvre pour gérer l'enchaînement de ses configurations. En effet, les commutations des interrupteurs dépendent de la commande qu'on leur impose, mais aussi du passage par zéro des courants les traversant et/ou de la tension à leurs bornes. Pour décrire l'enchaînement des configurations de ce bloc, il faut donc commencer par calculer ces grandeurs secondaires qui dépendent directement des grandeurs principales calculées dans le système global (courants et tensions en amont et en aval du commutateur). Ensuite, on peut mettre en oeuvre les nombreux tests de commutation des interrupteurs, c'est à dire la gestion de l'enchaînement des configurations du commutateur.

3) RESULTATS

Ce problème comportant des équations de taille importante, nous avons rejeté les réorganisations donnant les formes [1] et [2] au profit de celle fournissant la forme [3]. Ce choix correspond à un compromis entre la taille du programme généré, le temps d'exécution de notre générateur, et ce que nous souhaitons faire des équations réorganisées (ici : uniquement de la simulation).

Cette application a été étudiée au laboratoire et programmée manuellement, en utilisant le logiciel MC2, alors que notre générateur n'existait pas encore [PAZ92]. Ainsi, nous avons pu comparer les performances de notre approche avec l'utilisation de notre générateur, par rapport à une utilisation directe d'un logiciel classique du type MC2.

Grâce à notre générateur, sur cette application, nous avons obtenu :

- un gain considérable en temps de développement du programme de simulation final, pour la partie puissance électrique de l'association (dans un rapport supérieur à 10),
- une grande fiabilité des calculs formels de réorganisation des équations (risques d'erreurs très minimisés),
- la réduction automatique des systèmes d'équations d'état.

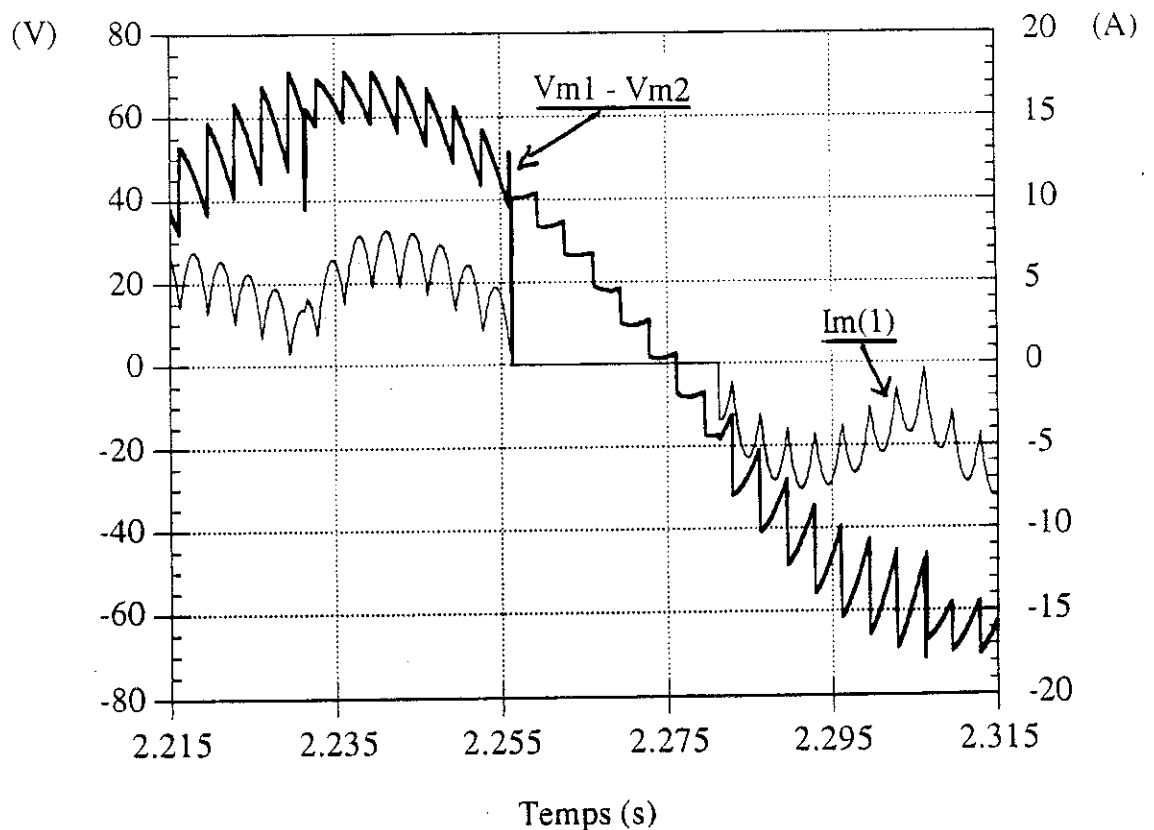
Notons que sur PC 80-386, 33 MHz, pour la partie puissance électrique, notre générateur ne met que dix heures pour effectuer son travail, lorsque l'on souhaite obtenir la forme [3] pour les équations de chaque configuration de l'association.

L'obtention de la forme [2] accroît le temps d'exécution du générateur dans un rapport 17.

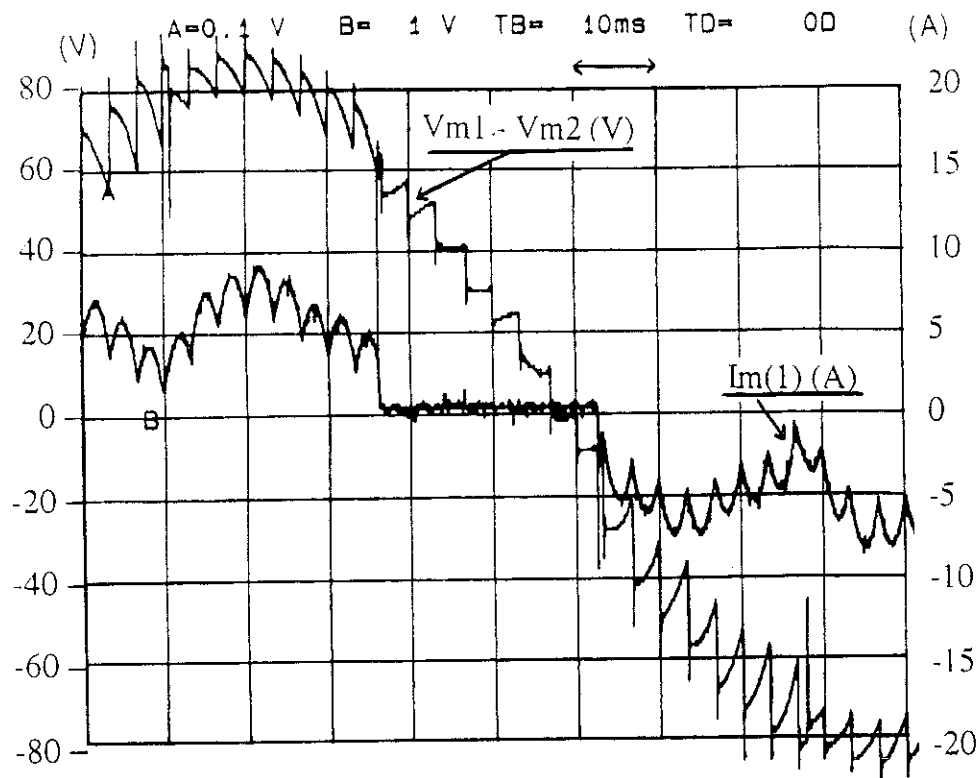
Grâce au banc d'essai du laboratoire, nous avons vérifié nos résultats de simulation avec des résultats expérimentaux (cf figure IV.8).

Il est important de noter que, dans ce cas, la forme [3] permet d'obtenir un code informatique très clair par rapport aux autres formes [1] et [2] qui sont trop lourdes. En effet, ces dernières peuvent donner des formules de plus de 500 lignes dans le format de programmation Fortran !

Remarque : La programmation réalisée par notre générateur, pour MC2, est très claire et facilement compréhensible par tout utilisateur habitué à employer MC2 sans notre générateur.



Résultats de simulation



Résultats expérimentaux

Figure IV.8 : Résultats de simulation : comparaison avec l'expérience

C. BILAN SUR NOTRE APPROCHE EN SIMULATION

1) INTERFACE UTILISATEUR ET PORTABILITE

Nous n'avons pas réalisé d'interface utilisateur interactive dans le cadre de cette thèse. En effet, c'est un travail important sur lequel des études d'ergonomie doivent être réalisées, et ces interfaces ne sont pas nécessaires pour valider notre approche.

De plus, l'approche non interactive par l'utilisation de fichiers permet d'offrir à notre générateur une grande portabilité. En particulier, cela permet d'envisager son couplage éventuel avec d'autres outils d'analyse (outils de simulation, systèmes experts...). Il en est de même pour les équations réorganisées que notre générateur peut fournir sous forme de fichiers en C ou en Fortran.

2) LA NECESSITE DE CONNAITRE UN LANGAGE DE PROGRAMMATION

La description du séquençement des boucles de commande numérique requiert des connaissances dans un langage de programmation (ici : Fortran). Cependant, ces connaissances restent très rudimentaires. Il en est de même du format des grandeurs de l'AMCC à simuler (ici : format Fortran) (cf annexe III). Il peut sembler regrettable que l'on ne puisse pas passer outre, mais cela ne représente qu'une faible partie du programme de simulation final. De plus, ceci ne limite nullement l'approche que nous proposons, notamment au niveau de la description modulaire de la boucle de commande numérique.

Remarque : le séquençement de la commande pourrait être décrit par une interface utilisateur du type GRAFCET, plutôt que du type algorythmique.

3) AVANTAGES

Nous allons maintenant résumer les points forts de notre approche en simulation.

Notre démarche de simulation est très structurée. En fait, la rigueur de description des problèmes, demandée à l'utilisateur, pourrait sembler très contraignante, mais elle permet :

- de limiter la quantité d'informations à fournir au générateur,
- de rendre concis et facilement compréhensibles, les fichiers de données fournis au générateur et les sous-programmes qui correspondent aux blocs.

Pour toute AMCC, notre générateur réalise la majeure partie de la programmation informatique nécessaire pour obtenir un programme de simulation. Seule la programmation du séquençement de la boucle de commande numérique reste à la charge de l'utilisateur.

Le formalisme de description des équations de commande analogique et de puissance est très naturel et sans aucun a priori sur la façon dont elles sont réorganisées et résolues.

Il n'y a pas d'erreur de programmation ou de calcul analytique dans le code informatique produit par notre générateur. Cette programmation automatique est claire, structurée et facilement accessible ou modifiable par un utilisateur familier de l'emploi direct du moteur de résolution (ici MC2).

Un gain considérable sur le temps de développement d'une nouvelle application de simulation est obtenu, par rapport aux outils classiques où le niveau de modularité des associations reste parfois très limité.

Notre générateur peut être adapté à d'autres logiciels de résolution d'équations différentielles que MC2. Nous avons essayé ceci en partie pour le logiciel OSSE (développée par O. Normand en C++, au sein du Laboratoire [NOR92]).

Les modifications de structure d'une application de base se font très rapidement. Pour la puissance et la commande analogique, il suffit simplement de modifier le fichier de données destiné au générateur. Pour la commande numérique, il faut modifier le séquençement des appels des sous-programmes correspondant aux blocs TS_n .

Il est possible de faire différentes analyses (études), à partir des équations complètes de l'AMCC étudiée, si elles sont réorganisées sous la forme [1].

4) BILAN PAR RAPPORT AUX OUTILS DE SIMULATION CLASSIQUES D'AMCC

Le tableau de la figure IV.9 résume les différents aspects de notre approche par rapport aux outils de simulation classiques.

Notre générateur permet d'optimiser la réorganisation des équations, de faire des tests de cohérence et de validité car nous l'avons créé dans un objectif précis : l'étude des AMCC.

Afin que notre outil ne soit pas trop complexe et difficile à maîtriser et à maintenir, nous l'avons conçu dans un langage de haut niveau : Macsyma.

Dans le même objectif, notre générateur est construit de façon modulaire. Chaque module a sa propre fonction, par exemple :

- trier les équations,
- réorganiser les équations,
- simplifier les équations,
- réaliser les sous-programmes d'interface utilisateur du programme de simulation final,
- etc...

Cette modularité facilite beaucoup la maintenance de notre outil.

	Outils de simulation classiques	Notre approche
Niveau de décomposition modulaire des associations	limité essentiellement par la causalité existant entre les différents blocs <i>cas de la partie puissance électrique de l'application traitée au chapitre IV.B</i>	pas de limitation
Création de blocs pour les parties de puissance et de commande analogique	travail complexe s'il est impossible de décomposer l'association étudiée en de nombreux blocs, c'est à dire, si on est obligé de construire de gros blocs (approche globale dans les blocs). Nécessité de décrire les équations en explicitant leur résolution. <i>cas de la partie puissance électrique de l'application traitée au chapitre IV.B</i>	simple , description des équations sans a priori sur la façon dont elle sont résolues
Travail de programmation informatique	important	faible, rudimentaire
Temps de développement des parties de commande analogique et de puissance d'une nouvelle application	très important <i>cas de la partie puissance électrique de l'application traitée au chapitre IV.B</i>	faible : le générateur fait le travail
Fiabilité des programmes de simulation obtenus	risque d'erreur non négligeable s'il y a de gros calculs analytiques à réaliser : <i>cas de la partie puissance électrique de l'application traitée au chapitre IV.B</i>	risque d'erreur très faible pour les calculs analytiques et la programmation réalisée par le générateur
Connaissance de la structure de l'outil de simulation pour l'utilisateur	nécessaire	nulle
Exploitation de l'outil	uniquement simulation	simulation et utilisation des équations réorganisées formellement
Points semblables	- séquençement de la commande numérique - création de blocs pour la commande numérique - temps de développement de la partie commande numérique d'une nouvelle application	

Figure IV.9 : Bilan comparatif entre notre approche avec une approche classique

CONCLUSION

A la fin de cette étude, nous disposons d'un ensemble d'outils (générateur de programmes associé à un moteur de résolution) pour la simulation des AMCC. Sa souplesse d'utilisation et ses performances nous permettent d'envisager de l'intégrer dans un ensemble plus vaste, destiné à aider le concepteur d'AMCC dans différentes étapes de la conception.

Pour cela, cet outil devra être associé à d'autres moyens d'étude : expérience et autres outils d'analyse (simulation, calcul...). Nous nous proposons de présenter ce système d'aide à la conception dans les chapitres suivants.

Chapitre V

**Modélisation du processus de conception des
associations machine convertisseur commande**

CHAPITRE V

MODELISATION DU PROCESSUS DE CONCEPTION DES AMCC

Dans ce chapitre, nous présentons les différents concepts qui nous ont permis de modéliser et de systématiser certaines étapes du processus de conception des AMCC à l'aide d'un système expert. Dans un premier temps, nous proposons une démarche de conception (figure V.1).

A. DEMARCHE DE CONCEPTION PROPOSEE

Dans cette démarche, on retrouve les principales étapes du processus de conception décrit au chapitre I.A. Cependant, nous distinguons ici trois grandes phases :

- 1) **Choix "grossier" des structures** des AMCC qui répondent au cahier des charges (défini au début du processus), puis évaluation qualitative des performances et des propriétés liées à ces choix.
- 2) **Dimensionnement et définition détaillée** des composants de chaque structure retenue.
- 3) **Analyse** de ces structures par la simulation et la réalisation de **prototypes**. Dans cette dernière phase, le concepteur peut chercher à améliorer et à optimiser certaines solutions; il peut donc être conduit à reprendre la phase précédente (2).

Nous avons entièrement automatisé la première phase à l'aide d'un système expert.

Cette démarche de conception est caractérisée par le fait que plusieurs solutions sont traitées **simultanément et en parallèle**. C'est-à-dire que plusieurs expertises sont confrontées pour obtenir la ou les meilleures solutions répondant au cahier des charges.

Afin de systématiser certaines étapes de ce processus de conception à l'aide d'un système expert, nous avons dû structurer la connaissance utilisée dans celui-ci, comme nous le présentons maintenant.

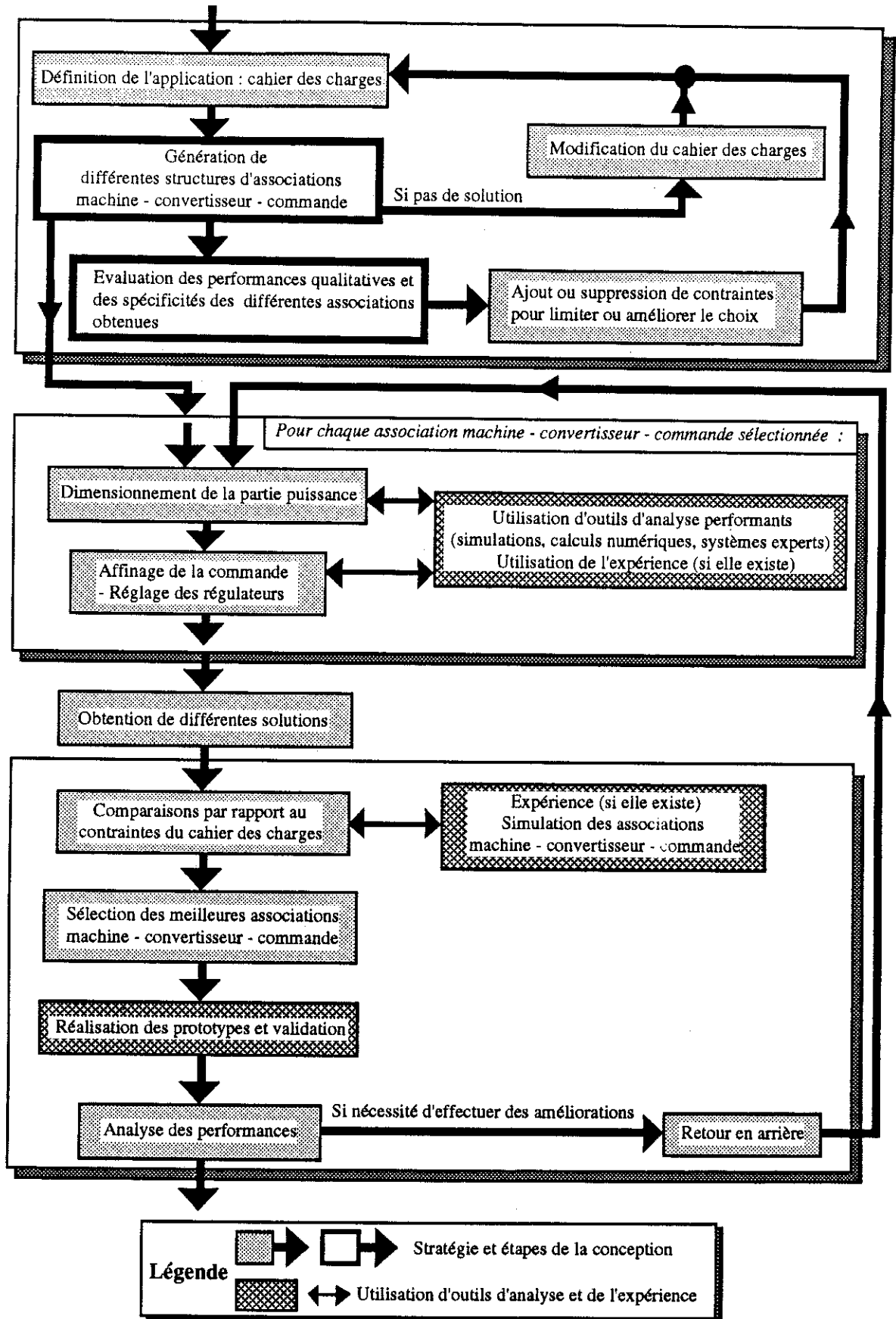


Figure V.1 : Stratégie de conception proposée

B. MISE EN FORME DE LA CONNAISSANCE

1) ARCHITECTURE DU SYSTEME EXPERT

Nous avons utilisé l'architecture présentée au chapitre I.D pour élaborer notre système expert (figure V.2).

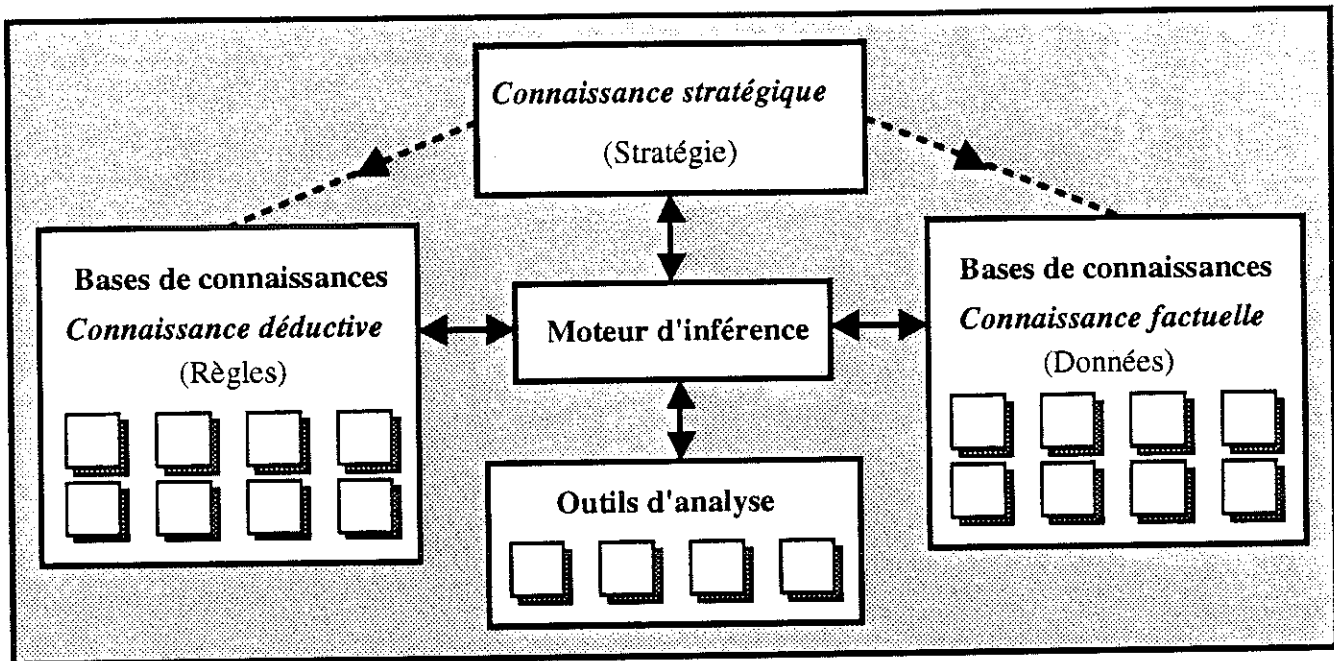


Figure V.2 : Architecture d'un système expert à bases de connaissance

Dans celui-ci, la stratégie apparaît en tant que telle pour atteindre un but précis (lorsqu'elle est guidée par les buts), ou elle peut aussi être contenue dans des règles (lorsqu'elle est guidée par les faits).

Les règles d'expertise s'appliquent sur les données selon la stratégie qui est décrite dans le système expert.

Cela nous conduit à modéliser trois types de connaissances (figure V.2) dont nous présentons maintenant les différents formalismes que nous avons utilisés pour celles-ci.

2) LA CONNAISSANCE FACTUELLE

La connaissance factuelle est stockée dans des bases de données. Pour assurer la possibilité d'évolution de ces bases, et en raison de la complexité et du nombre important de données qu'il faut manipuler dans le processus de conception des AMCC, nous avons dû structurer ces

données sous forme abstraite et condensée. Ainsi, nous les avons décrites dans le formalisme **objet** [MEY91] que nous allons présenter.

a) Le formalisme objet

Pour représenter des éléments ayant la même structure, les mêmes propriétés et les mêmes caractéristiques, nous utilisons le concept générique de **classe**. Les éléments réels d'une classe sont appelés **objets**.

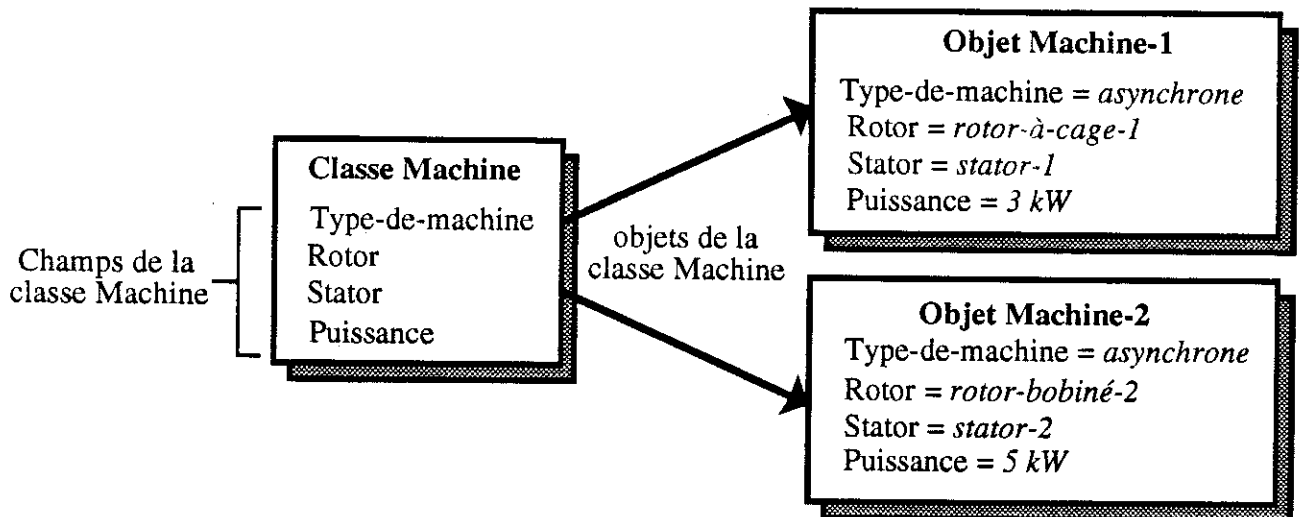


Figure V.3 : Liens entre classes et objets

Nous appelons **champs**, les différentes propriétés et caractéristiques des classes, donc des objets.

Par exemple, la classe "machine" permet de caractériser l'ensemble des machines (figure V.3), par leurs propriétés et leurs caractéristiques (champs), par exemple : rotor, stator, puissance, etc... Une machine réelle, quant à elle, est représentée par un objet dont la structure est donnée par la classe "machine". Les valeurs de ses propriétés et de ses caractéristiques correspondent à celles de l'élément physique qu'il représente.

Les propriétés et caractéristiques peuvent être de plusieurs types :

- un objet (ex : stator-1, rotor-1),
- un nombre entier ou réel (ex : puissance),
- un symbole (ex : type de machine),
- une liste des possibilités précédentes.

A cette notion générique de classe, est associée la notion d'**héritage**. Nous utilisons cette dernière lorsqu'à l'intérieur d'une même classe, nous souhaitons définir des sous-classes ayant les propriétés et les caractéristiques de la première classe (appelée alors "classe mère"), ainsi que d'autres propres à chacune d'elles. Cette notion d'héritage permet d'alléger fortement la

description des classes et des sous-classes, et de définir différents niveaux de décomposition des données. Nous avons, par exemple, décomposé la classe "convertisseur-statique" en deux sous-classes (figure V.4) :

- la classe "convertisseur-direct" (1),
- la classe "convertisseur-indirect" (2).

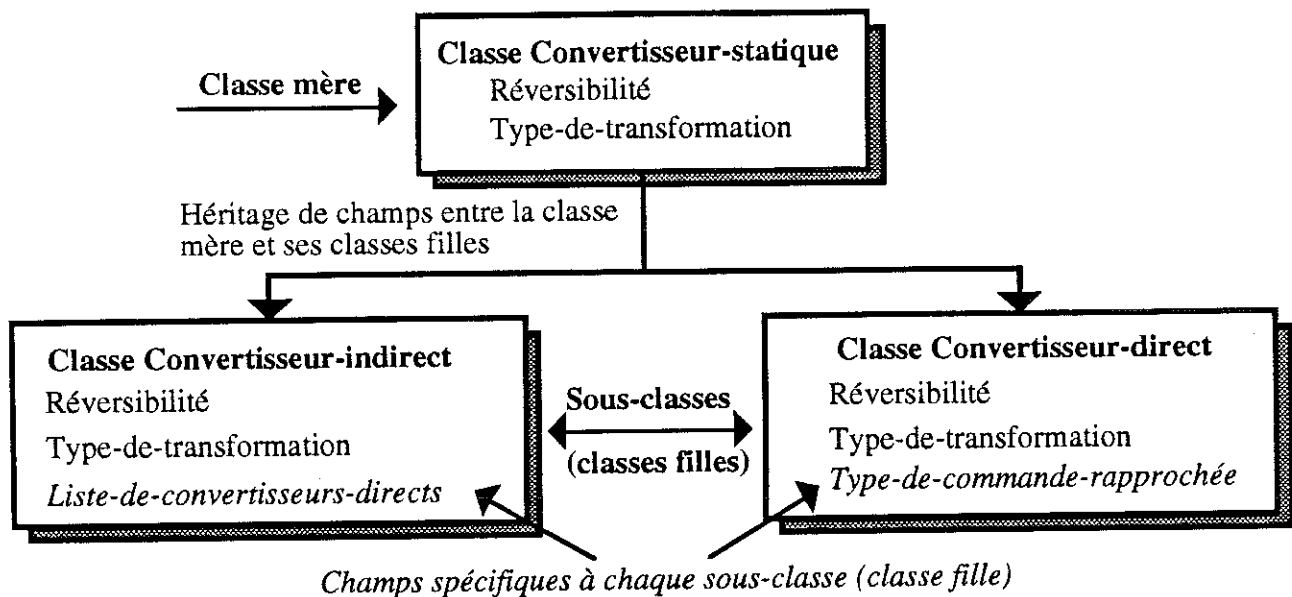


Figure V.4 : Héritage entre classe mère et classe fille

Les classes filles héritent de toutes les propriétés et caractéristiques de leur classe mère, par exemple : réversibilité, type de transformation. Cependant, elles possèdent chacune leurs propres propriétés et caractéristiques, par exemple : type-de-commande-rapprochée (1), liste-de-convertisseurs-directs (2).

Pour mettre en oeuvre les données dans ce formalisme, une des principales difficultés est de savoir ce qui doit être un objet ou un champ (propriété ou caractéristique) d'objet. Dans ce but, nous avons défini des critères de décomposition des données.

b) Critères de décomposition des données

Pour structurer les données, nous avons imposés différents critères. Ainsi, nous représentons une donnée par un objet si :

- il y a au moins deux manières de la qualifier ou de la caractériser,
- elle peut être décomposée en plusieurs données,

- elle est utilisée dans de nombreux champs de classe.

Par exemple, nous avons défini une classe "puissance", car nous avons besoin de caractériser cette donnée de différentes façons. Celles-ci ont été formulées sous forme de propriétés de la classe "puissance" :

- puissance-maximale,
- puissance-minimale,
- puissance-quantitative,
- puissance-qualitative.

c) Modélisation de l'imprécis

Certaines données sont imprécises, c'est-à-dire qualifiées par des termes du type : fort, faible, petit, grand, moyen, très fort ... Dans notre système, nous avons dû établir un lien entre ces données "floues" et les données quantitatives.

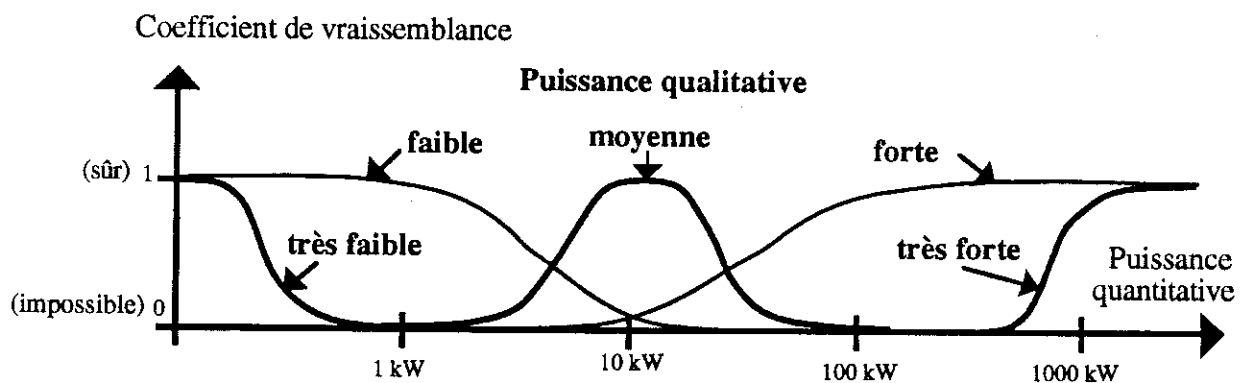


Figure V.5 : Modélisation possible d'un nombre imprécis

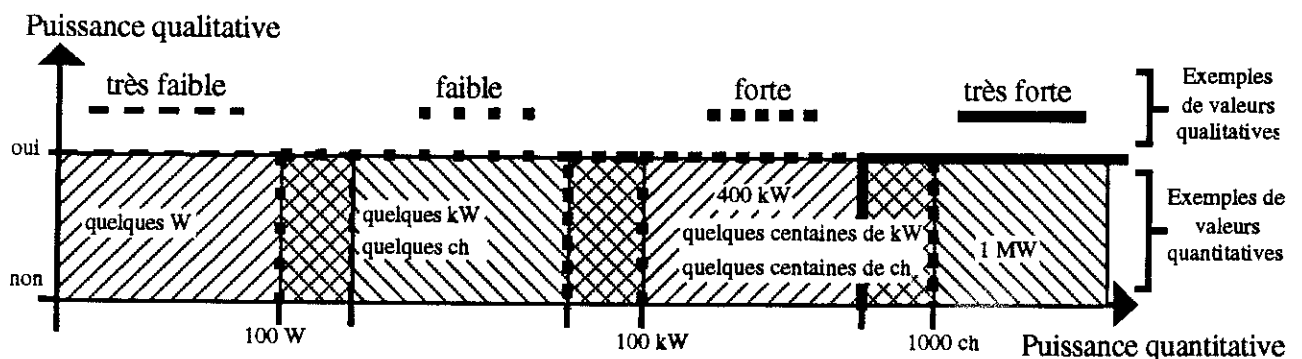


Figure V.6 : Lien entre les valeurs de la puissance quantitative et celles de la puissance qualitative utilisées dans les règles

Pour réaliser ceci, il est possible d'employer la logique floue [KAU87] [HAT91] [FRA91]. Cette dernière propose d'utiliser des fonctions qui, pour chaque terme ou proposition flou, varient entre 0 (impossible) et 1 (certain) et qui permettent ainsi d'effectuer le lien avec des grandeurs quantitatives. La figure V.5 montre un exemple de représentation des grandeurs qualitatives d'une puissance. Cependant, pour ne pas avoir à résoudre ce problème complexe de modélisation en même temps que celui de la représentation des règles déductives, nous avons décidé de prendre un modèle moins précis, mais plus rigide et facile à contrôler pour représenter de telles données. La figure V.6 montre le modèle que nous avons choisi pour caractériser la puissance des AMCC dans les objets et les règles. Nous avons fait chevaucher les différentes zones qualitatives au niveau de leur jonction. Les correspondances entre les valeurs qualitatives et quantitatives se font en tout ou rien. Par exemple, une puissance est faible, ou faible et forte, ou forte, ou ..., mais aucune nuance n'est mise en oeuvre pour dire qu'elle est faible avec une vraisemblance de 50 %.

Remarque : Dans des travaux ultérieurs, ce type de considération pourra être pris en compte.

La base de données est modifiée au cours du processus de conception par les règles qui sont appliquées. Ces dernières constituent la connaissance déductive que nous présentons maintenant.

3) LA CONNAISSANCE DEDUCTIVE

Pour constituer et structurer une base de connaissance, nous avons utilisé la démarche que nous développons ici.

a) Acquisition des règles

a.1) Acquisition de l'expertise

La connaissance peut être acquise par différentes sources (cf figure V.7). Dans le domaine de la conception des AMCC, cette connaissance est très vaste, nous n'avons donc pas pu tout recueillir. Toutefois, nous avons veillé à utiliser des sources d'expertise complémentaires, de manière à couvrir les différents aspects, et problèmes, liés à la conception des AMCC :

- entretiens avec un expert,
- livres [MOH89] [BOS86] [SEG85],
- recueils de formation [INP92A],
- cours [INPG92B],

- notice et cahier des charges de produits industriels.

Remarque : Cette multiexpertise n'a pas fourni de contradiction sur une même connaissance ; et elle s'est traduite par l'obtention de plusieurs solutions pour un même problème.

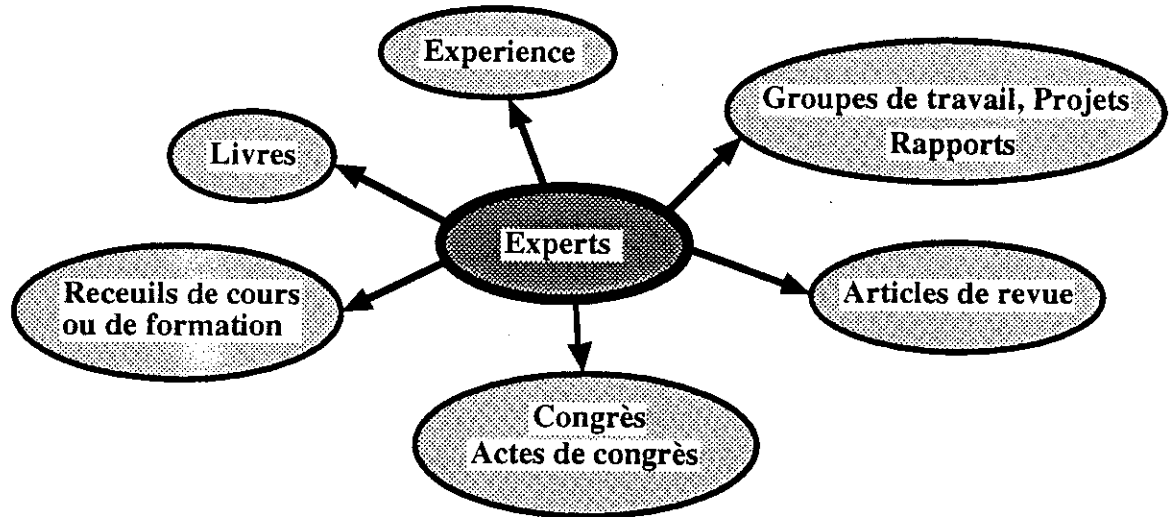


Figure V.7 : Différentes sources de connaissance existantes

Malheureusement, l'ensemble de ces connaissances a été recueillie sous la forme de texte, dans un formalisme brut et confus, par exemple :

- "Si on alimente un moteur asynchrone par un onduleur de tension en "MLI"... les harmoniques engendrées (par la MLI) génèrent des pertes plus élevées qu'en "carré"... Ce phénomène est très fortement lié à la nature du moteur : propriétés magnétiques, fréquence de commutation. En raison de ces pertes, il faut utiliser un moteur d'une puissance nominale 5 à 10 % supérieure à celle souhaitée".
- "Pour des applications bas de gamme, on peut utiliser une commande pleine tension, avec une machine trapèze, ce qui équivaut à une machine à courant continu. Dans ce cas, pas de "MLI", mais une commande en tout ou rien de chaque phase, donc un commutateur de tension".

Nous avons donc dû analyser et restructurer cette connaissance.

a.2) Analyse de l'expertise - classement des règles

Nous avons donc repris les différents textes recueillis afin d'extraire et de formaliser des règles d'expertise.

Pour faciliter ce travail, nous avons réalisé une fiche pour chaque règle extraite, que nous avons classée (figure V.8) :

- par type de moteur :
 - moteur pas à pas,
 - moteur à courant continu,
 - moteur synchrone,
 - moteur asynchrone,
 - généralités (communes aux quatre types de moteurs);
- par source d'expertise :
 - entretien avec ...,
 - livre de ...,
 - recueil de formation de ...,
 - cours de ...,
 - notice de ...

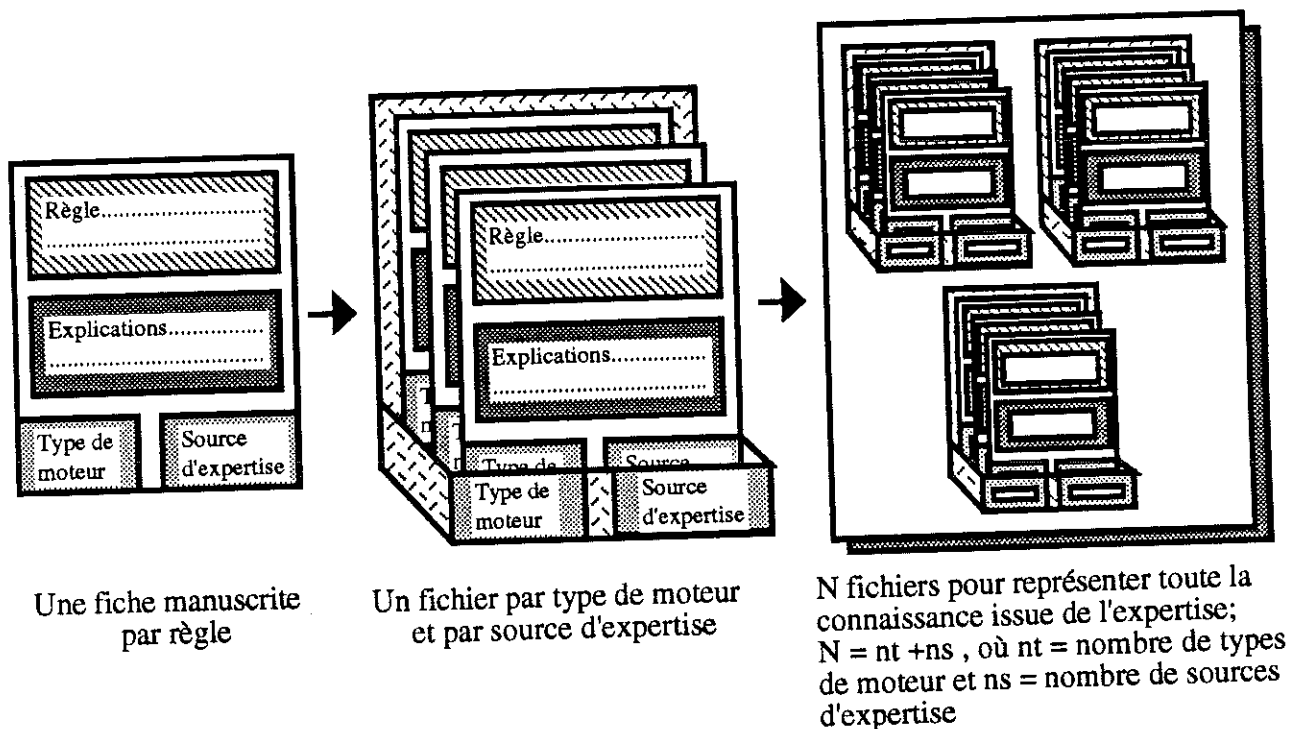


Figure V.8 : Acquisition de la connaissance

Dans ce classement, ces informations sont synthétisées sous la forme d'une étiquette, avec un numéro de règle, pour chaque fiche. Ce classement est très important pour pouvoir remonter d'une règle implantée dans le système expert à son origine.

Remarque : Notons que chaque fiche est munie d'une explication de règle.

Nous avons alors analysé puis reformulé l'ensemble des règles, de manière à les rendre le plus explicite possible et en veillant à ce qu'elles soient cohérentes entre elles. En effet, nous avons dû nous assurer qu'une même caractéristique, ou propriété, portait le même nom dans toutes les règles, par exemple : "commande dq" s'écrit aussi "commande dans le référentiel dq", ou "commande vectorielle dq"...

Grâce à cette analyse, nous avons pu reformuler les règles dans le formalisme que nous nous sommes imposés : la règle de production.

b) La règle de production

Rappelons que ce formalisme est du type "si condition alors action" par exemple :

Si un moteur asynchrone est alimenté par un onduleur de tension commandé en MLI **alors**

- pertes fer élevées dans le moteur,
- déclassement du moteur de 5 à 10 %,
- harmoniques de tension.

Ce formalisme est assez contraignant, mais nous l'avons choisi car il est simple, rigoureux et facile à maîtriser. Ainsi, il nous a permis de tester efficacement la stratégie de conception que nous avons mise en oeuvre.

c) Modélisation de l'incertain

Certaines connaissances sont caractérisées par des nuances telles que : préférable, plutôt, il suffit de, souvent, parfois... Celles-ci peuvent être modélisées par un coefficient de vraisemblance variant de 0 (impossible) à 1 (certain) pour chaque règle de production [HAT91]. Par application successive des règles, on obtient des solutions avec une vraisemblance comprise entre 0 et 1. Cependant, ceci nécessite l'emploi de techniques issues de la modélisation floue [KAU87] [HAT91]. Pour ne pas avoir à résoudre simultanément plusieurs problèmes de modélisation, nous avons décidé (comme au paragraphe V.B.2.c) de repousser ce type d'étude à une phase ultérieure. Ainsi, pour modéliser ces nuances, nous avons décidé que le système expert doit donner une solution dès qu'elle a une chance de pouvoir exister.

Les règles sont appliquées sur les données, suivant une stratégie qui doit être mise en oeuvre dans le système expert, comme nous le montrons maintenant.

4) LA CONNAISSANCE STRATEGIQUE

A partir de l'expertise recueillie, nous avons pu systématiser et assister le concepteur dans certaines étapes du processus de conception. Pour assurer la maintenance et l'évolution du système expert, nous avons réparti les règles en différents groupes facilement accessibles et modifiables, que nous appelons **tâches**.

a) Les tâches

a.1 Caractéristiques

Fonctions des tâches	Eléments concernés par ces fonctions
Critères de définition et de dépendance	application / alimentation - commande / couplage charge - moteur / moteur / capteur de vitesse / commande
Règles de choix	moteur / alimentation - commande / circuits auxiliaires / capteurs
Détails pour préciser la structure et détails de conception	commande
Propriétés caractéristiques	moteur / variateurs de vitesse électronique / ensemble machine - convertisseur / convertisseurs / ensemble machine - convertisseur - commande / capteurs
Dimensionnement	convertisseurs / moteur / régulateur - commande / ensemble machine - convertisseur / ensemble machine - convertisseur - commande / capteurs / inertie
Améliorations	moteur / alimentation (convertisseurs) / commande / capteurs
Comparaisons	moteur / alimentation (convertisseur) / commande / capteurs
Performances	convertisseurs / commande / capteurs
Fonctionnements particuliers	convertisseurs / commande / freinage / blocage / démarrage / positionnement
Influences	de la commande sur le moteur / de la commande sur l'alimentation (et les convertisseurs) / des convertisseurs sur la commande

Figure V.9 : Fonctions possibles définies pour une tâche à partir de l'expertise recueillie

Nous caractérisons une tâche par :

- Une **fonction**, qui correspond à l'une des étapes de la stratégie de conception. Nous en avons définies plusieurs à partir de l'expertise recueillie, et qui sont répertoriées sur la

figure V.9. Cependant, cette liste n'est pas exhaustive.

- Un **but**, qui est d'affecter des propriétés et caractéristiques d'objets ou de créer des objets, en fonction du cahier des charges, de propriétés ou caractéristiques d'objets déjà affectées, et / ou d'objets existants.
- Un **mode de raisonnement**, qui est soit séquentiel, soit parallèle (cf § V.B.4.c).
- Un **ensemble de règles**, qui ont le même but et la même fonction.
- Un **mode d'exécution des règles**, qui peut être, par exemple, appliquer les règles dans un ordre quelconque mais une seule fois.

a.2 Critères de formation d'une tâche

Pour construire ces tâches, nous avons défini les critères suivants :

- Dans une tâche, les règles doivent toutes avoir le même niveau de contrainte dans leur partie condition (c'est-à-dire test). Ainsi, pour un même but et une même fonction, on peut avoir une tâche qui donne des solutions en fonction de critères très restrictifs (puissance de 5 kW, plage de vitesse de 1 à 50, application de levage ...) et une autre qui donne des solutions en fonction de critères généraux (contrôle de vitesse, contrôle de position ...).
- Pour chaque tâche, les règles peuvent s'appliquer dans un ordre quelconque et une seule fois sur chaque combinaison d'objets instanciables (c'est-à-dire vérifiant les conditions d'application des règles).

b) Les démarches de raisonnement

Dans chaque étape de conception que nous avons systématisée partiellement, ou totalement, le raisonnement que nous utilisons est en général **guidé par les buts**. Ceci signifie que la plupart des tâches s'exécutent dans un objectif précis, en fonction de la stratégie définie dans le système expert. Les règles de ces tâches s'appliquent en faisant abstraction du fait que les données nécessaires au raisonnement sont disponibles ou non.

Nous avons aussi dû définir des tâches **guidées par les faits** (les données). Dans ce cas, la stratégie qui permet d'activer leur exécution est codée dans des règles, et ces tâches ne

s'exécutent que si certaines données existent. Nous avons utilisé ce type de tâche pour compléter la base de données lorsque cette base s'avère incomplète, avant d'appliquer d'autres tâches guidées par les buts, .

c) L'arbre de raisonnement

Pour représenter le raisonnement appliqué par le système expert, nous utilisons la notion d'**arbre de raisonnement**. C'est un moyen simple et efficace pour permettre à l'utilisateur du système d'accéder d'une part aux solutions fournies par le système et d'autre part au raisonnement utilisé pour obtenir celles-ci.

Avant d'aborder l'exploitation de cet arbre, nous allons introduire quelques notions de base nécessaires pour une bonne compréhension de la suite de ce rapport.

c.1 Notions de base sur l'arbre de raisonnement

On appelle **contexte**, à un instant t de l'exécution du système, l'ensemble des objets qui existent à cet instant, ainsi que la valeur de leurs champs (propriétés ou caractéristiques).

On dit qu'une règle est **instanciable** si elle peut s'appliquer sur un ou plusieurs objets. Alors, lorsqu'elle s'exécute, deux phénomènes peuvent se produire.

- Soit elle engendre un nouveau contexte, en créant des objets, et / ou en supprimant des objets, et / ou en modifiant des valeurs de champs d'objets (figure V.10). Dans ce cas, le raisonnement est dit **séquentiel**.
- Soit elle engendre plusieurs contextes différents, en créant des objets et / ou en supprimant des objets, et / ou en modifiant des champs d'objets (figure V.11). Dans ce cas, le raisonnement est dit **parallèle**.

Dans notre système, le raisonnement donne un branchement séquentiel lorsque les règles sont du type :

"si condition alors
 action 1
 et action 2
 et action 3
 et"

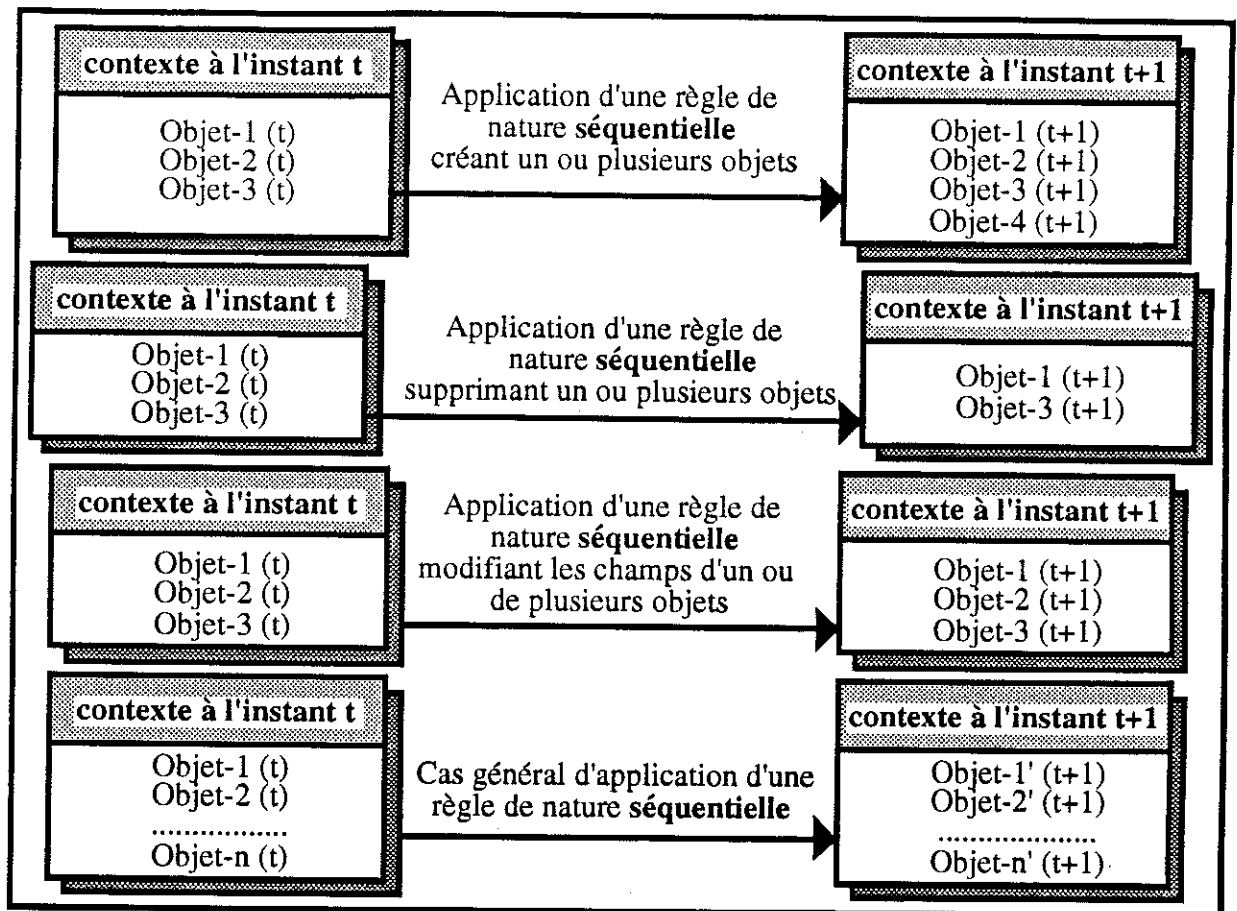


Figure V.10 : Raisonnement séquentiel

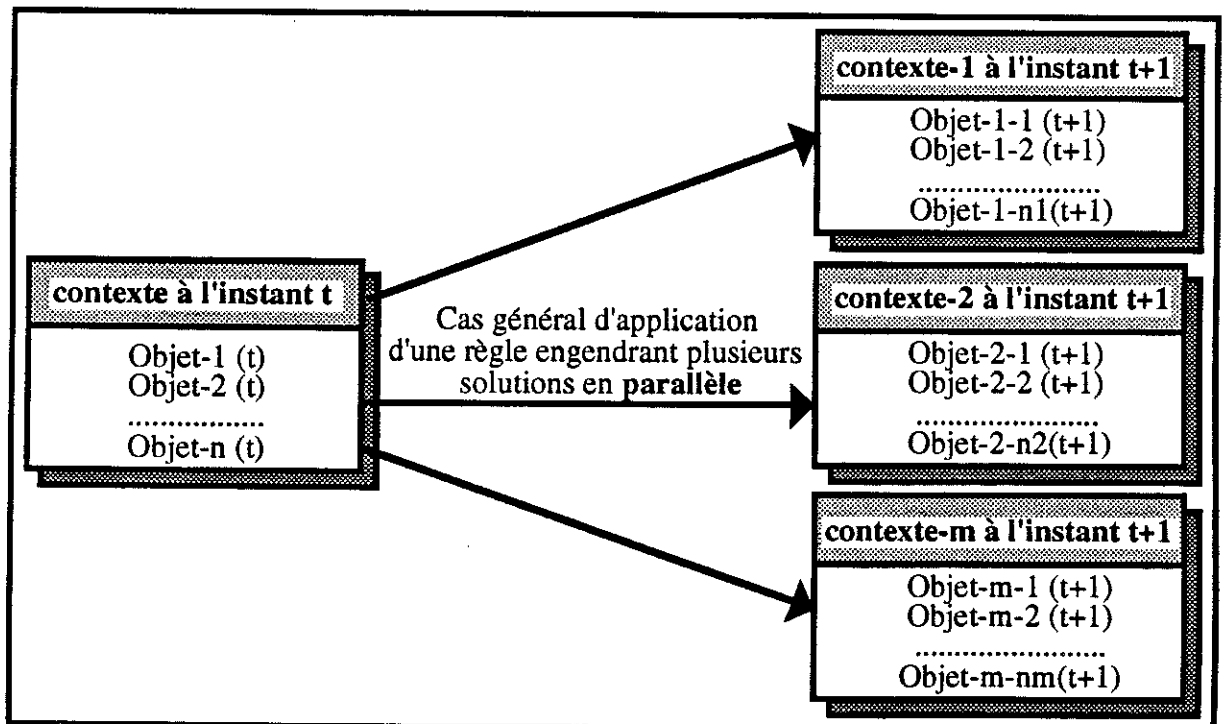


Figure V.11 : Raisonnement parallèle

et il donne un branchement parallèle lorsque les règles sont du type

"si condition alors

action 1

ou action 2

ou action 3

ou ..."

où chaque action engendre son propre contexte.

Par application successive des règles, on obtient un **arbre de raisonnement** qui ne contient qu'une seule branche si le raisonnement est purement séquentiel, mais qui peut en contenir plusieurs, si certaines phases du raisonnement sont parallèles (figure V.12).

c.2 Exploitation de l'arbre de raisonnement

Un arbre de raisonnement (figure V.12) est constitué de **feuilles** qui représentent chacune un contexte. Ces feuilles sont reliées entre-elles par une branche qui représente la règle qui a été appliquée sur le premier contexte pour donner le second.

Cet arbre visualise le raisonnement complet appliqué par le système expert, à partir du **contexte initial** (au début du raisonnement), pour donner les **contextes terminaux** (en fin de raisonnement ou à l'extrémité des branches).

Ces derniers sont dits **fermés** s'ils conduisent à un échec ou à une contradiction. Dans ce cas, plus aucun raisonnement ne peut s'effectuer à partir de tels contextes.

Sinon, un contexte terminal est dit **ouvert** lorsqu'un raisonnement peut se poursuivre à partir de lui.

Dès qu'une règle est applicable sur un ou plusieurs contextes terminaux, celle-ci s'applique successivement sur chacun d'eux et engendre à chaque fois un ou plusieurs nouveaux contextes terminaux.

Quand le raisonnement est terminé, on obtient les différentes solutions fournies par le système en consultant les contextes terminaux ouverts. Pour connaître le raisonnement qui a permis d'obtenir une solution, il suffit, à partir du contexte terminal où celle-ci est définie, de remonter l'arbre de raisonnement vers l'état initial, en consultant les règles qui ont été appliquées.

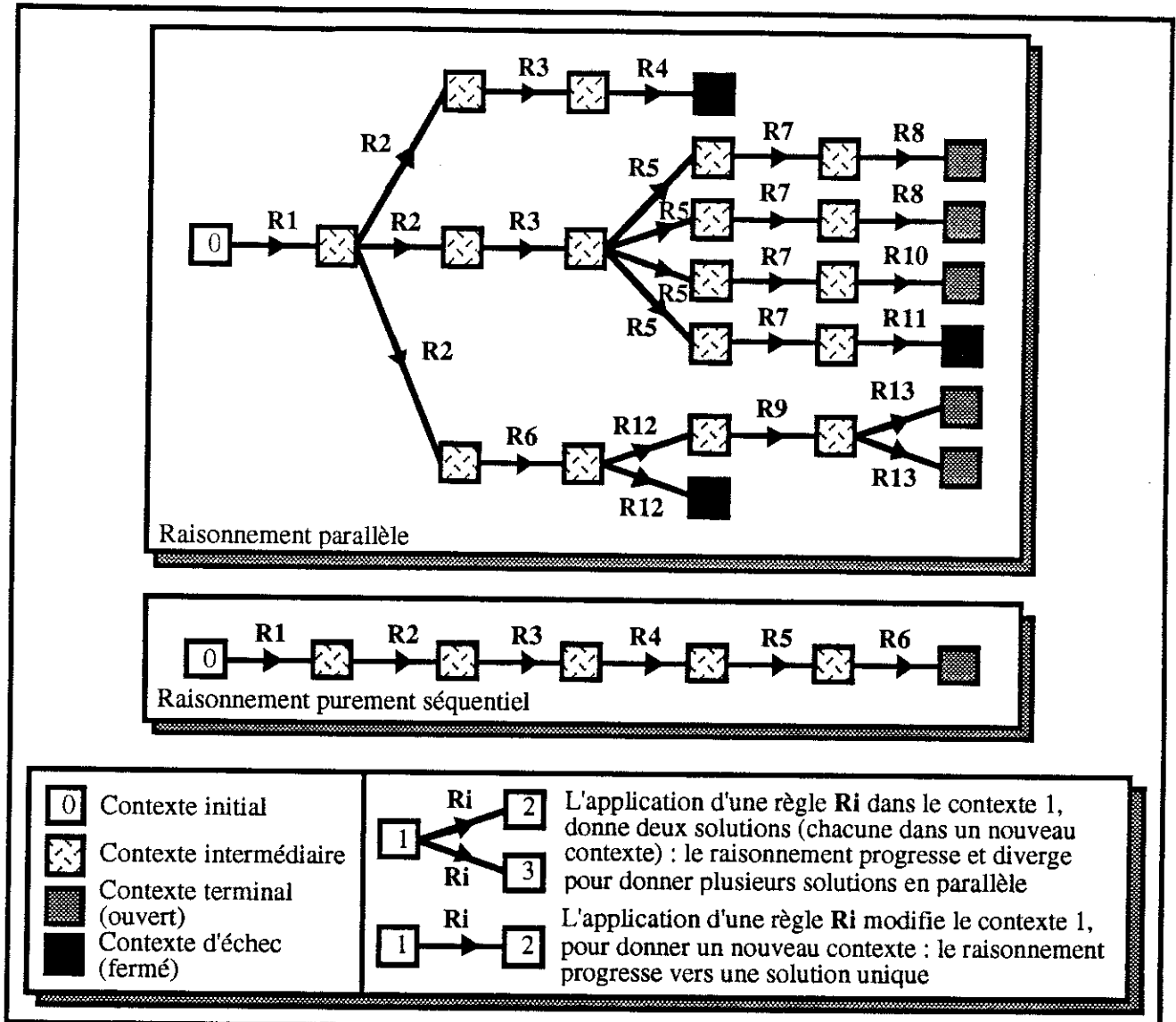


Figure V.12 : Arbres de raisonnement

C. UTILISATION D'OUTILS D'ANALYSE

Dans les différentes étapes de la stratégie de conception que nous proposons, il est nécessaire d'avoir recours à plusieurs moyens d'analyse :

- l'expérience,
- des outils de simulation numérique,
- des outils de calcul numérique,
- des systèmes experts de conception de machines [TRI91] ou de conception de convertisseurs statiques [AHR90] [CHH92]. Notons que l'emploi de l'outil de simulation développé au début de cette thèse permet de tester et de comparer différentes AMCC et différentes commandes de machine, avant de passer à l'expérience.

Dans notre système, l'utilisation de ces moyens d'analyse reste à la charge de l'utilisateur.

Remarque : un couplage automatique ou semi-automatique pourra être envisagé dans de futurs travaux entre le système expert et certains de ces outils, mais cela nécessitera un travail informatique important.

A partir de ce qui vient d'être présenté dans ce chapitre, nous avons pu élaborer les bases d'un système expert d'aide à la conception des AMCC que nous présentons dans le chapitre VI.

Chapitre VI

Mise en oeuvre du système expert

CHAPITRE VI

MISE EN OEUVRE DU SYSTEME EXPERT

A partir de l'étude et des concepts présentés au chapitre V, nous avons mis en oeuvre un système expert. Pour faciliter cette réalisation, nous avons utilisé un générateur de système expert : **Smeci** (Système Multi-Expert de Conception et d'Ingénierie) [SME92]. C'est une "boîte à outils" qui a fait ses preuves dans de nombreuses applications dont certaines au Laboratoire [GEN91].

Smeci offre les briques de bases sur lesquelles nous avons pu bâtir l'architecture de notre système. En particulier, il résout une bonne partie des problèmes purement informatiques que l'on rencontre dans tout système expert : gestion des bases de données, mise en oeuvre du moteur d'inférence, interfaçage de développement et d'utilisation ... Ainsi, son emploi nous a permis de ne nous préoccuper que des problèmes et concepts que nous souhaitons modéliser, et que nous présentons dans ce chapitre.

A. MISE EN OEUVRE DES DONNEES

1) LE FORMALISME DES OBJETS DANS SMECI

Smeci offre un formalisme et une interface interactive pour définir les classes et les objets (figure VI.1).

2) CONTRAINTES SUR LES OBJETS

Pour assurer une bonne cohérence entre les données manipulées et pour éviter une trop grande dispersion des valeurs pouvant être prises par les propriétés et les caractéristiques des objets, nous avons limité le nombre de ces valeurs.

Cette remarque concerne surtout les données qualitatives dont la valeur est un symbole (chaîne de caractères sans espace blanc). Par exemple, il existe plusieurs façons pour définir une "commande dq" :

- commande dans le référentiel dq,

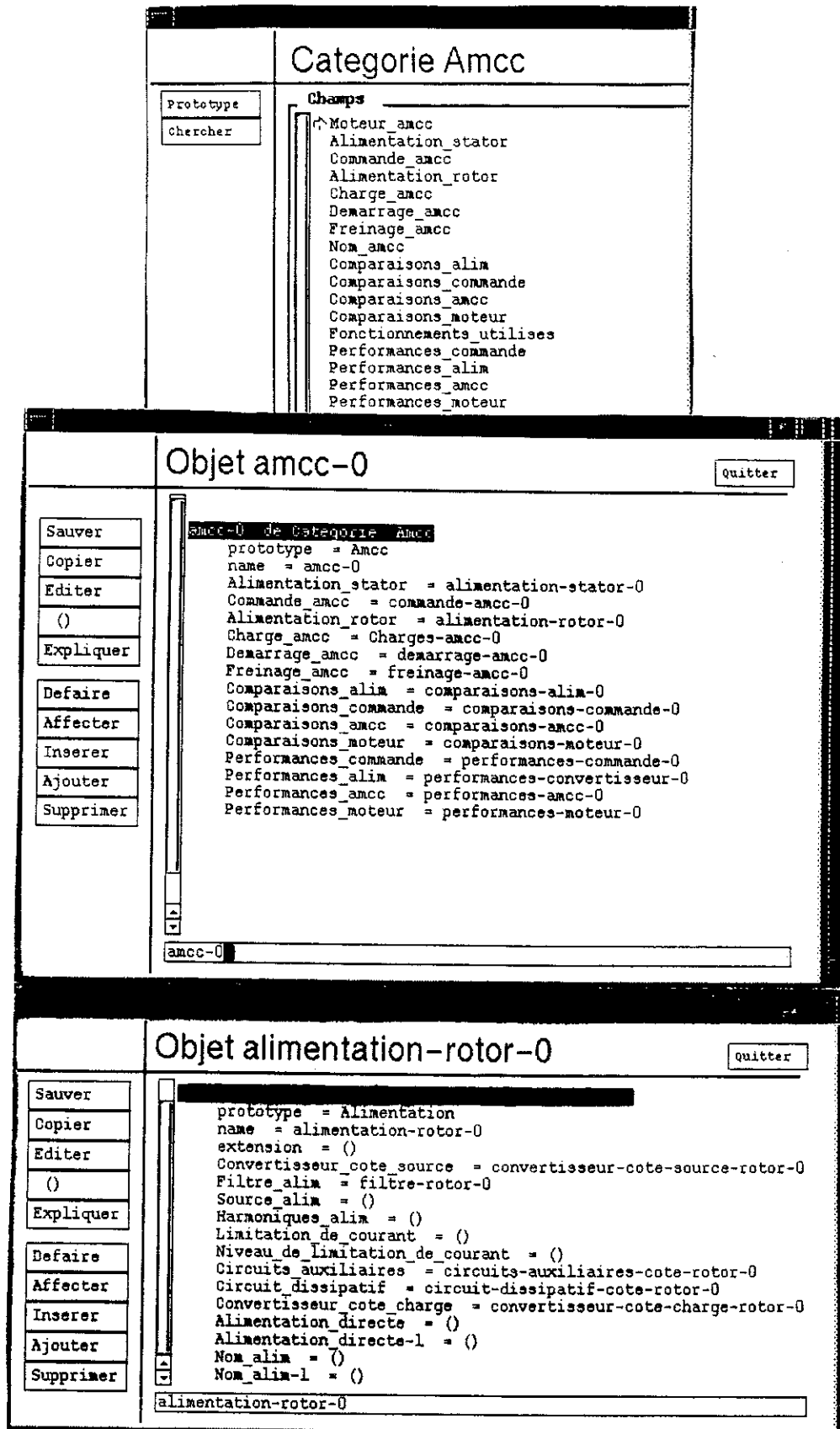


Figure VI.1 : Représentation des classes (catégories) et des objets dans Smeci

- contrôle du couple dans le référentiel dq,
- contrôle dynamique du couple dans le référentiel tournant dq,
- etc...

ou plus subtile : commande DQ !

Pour limiter les valeurs possibles des champs (propriétés ou caractéristiques) des objets, nous avons imposé plusieurs contraintes :

- les champs de type "nombre" sont caractérisés par leur format : réel ou entier,
- les champs de type "objet" ne peuvent appartenir qu'à une seule classe d'objets; par exemple, le moteur d'une AMCC solution doit appartenir à la classe "moteur" (ou à toute sous-classe de celle-ci),
- les champs de type "symbole" ne peuvent prendre que certaines valeurs; par exemple, la propriété "puissance-qualitative" de la classe puissance ne peut prendre que les valeurs suivantes :
 - très-faible,
 - faible,
 - très-forte,
 - forte,
 - () qui représente aucune affectation (vide).

Remarque : En général, nous n'avons pas contraint les champs de type "liste-de-symboles", car ils ne nous servent qu'à fournir des résultats de règles.

Par cette réalisation, nous avons pu mettre en oeuvre les règles et les tâches.

B . MISE EN OEUVRE DES REGLES ET DES TACHES

1) LE FORMALISME DES REGLES DANS SMECI

Pour décrire les règles, grâce à Smeci, nous avons pu utiliser un formalisme informatique interprété, donc très explicite (figure VI.2).

```

defregle GM_20a_alim_moteur_asynchrone_si_tres_forte_puissance_et_faible_vitesse
auteur lg
explication
fin-explication
soit A une Amcc et
    CC un Cahier_des_charges et
    M un Moteur parmi les Moteur_amcc de A et
    V une Vitesse parmi les Vitesse_souhaitée de CC et
    P une Puissance parmi les Puissance_souhaitée de CC
si Type_moteur^M = moteur_asynchrone et
    Vitesse_qualitative^V = faible et
    Puissance_qualitative^P = tres_forte
alors
    Nom_convertisseur^Convertisseur_cote_source^Alimentation_stator^A =
        cycloconvertisseur
    Alimentation_directe^Alimentation_stator^A = t
fin-regle

```

Figure VI.2. : Exemple de règles mise dans le format de Smeci

Dans une règle, on distingue plusieurs parties (figure VI.3).

```

auteur.....
explication.....
fin-explication
soit.....
si.....
alors.....
action.....
creer.....
fin-regle

```

Figure VI.3. : Formalisme de représentation des règles dans Smeci

a) L'affichage du raisonnement

La partie "**explication ... fin explication**" permet d'expliquer à l'utilisateur les opérations effectuées par la règle lorsqu'elle s'applique. Ces explications apparaissent à l'écran au cours de l'exécution du système expert.

b) L'instantiation

A un instant donné du raisonnement, une règle est "**instanciable**" (c'est-à-dire applicable) si sa partie condition est vérifiée. Ces tests se font à deux niveaux :

- "**soit ...**" : permet de s'assurer que les objets instanciés (c'est-à-dire auxquels la règle fait appel) existent dans le contexte courant où on veut appliquer la règle,
- "**si ...**" : sert à effectuer des tests sur les valeurs des propriétés des objets dont l'existence est testée dans le "soit ...".

c) Action

Si une règle est instanciable, alors les actions décrites dans la partie "**alors ... fin-règle**" peuvent s'exécuter. Celles-ci peuvent apparaître sous trois formes :

- affectation ou modification de propriétés d'objet, juste après "**alors ...**",
- exécution de procédures écrites en langage LISP [LIS87], qui modifient des objets ou activent l'exécution de tâches ou de calculs numériques ..., juste après "**action ...**",
- création de nouveaux objets, juste après "**créer ...**".

Après avoir implanté les règles dans le système, nous les avons regroupées en tâches suivant les critères présentés au chapitre V.

2) MISE EN OEUVRE DES TACHES

a) Formalisme des tâches dans SMECI

Dans Smeci, chaque tâche est associée à une base de règles qui contient les règles de cette tâche. Cette base est paramétrable dans le but de gérer :

- l'**instanciation** des règles (application d'une ou de toutes les règles sur une ou toutes les combinaisons d'objets instanciables),
- le **mode d'exécution** des règles (dans l'ordre ou non),
- le **mode de branchement** des règles (par rapport à l'arbre de raisonnement : parallèle ou séquentiel).

En annexe VIII, nous développons ces notions et nous donnons les détails techniques des combinaisons de ces paramètres que nous utilisons.

b) Cas particulier de règles représentées par une tâche

Le formalisme de description des règles de Smeci, n'offre pas la possibilité de décrire directement les règles contenant un "**ou**" dans leur partie action. Nous avons alors dû fournir un mode de description de ces règles pour qu'elles puissent effectuer un raisonnement parallèle. Ainsi, nous créons une tâche pour chacune de ces règles. Par exemple, la règle "si condition alors action 1 ou action 2" est transformée en une tâche avec deux règles s'appliquant en parallèle :

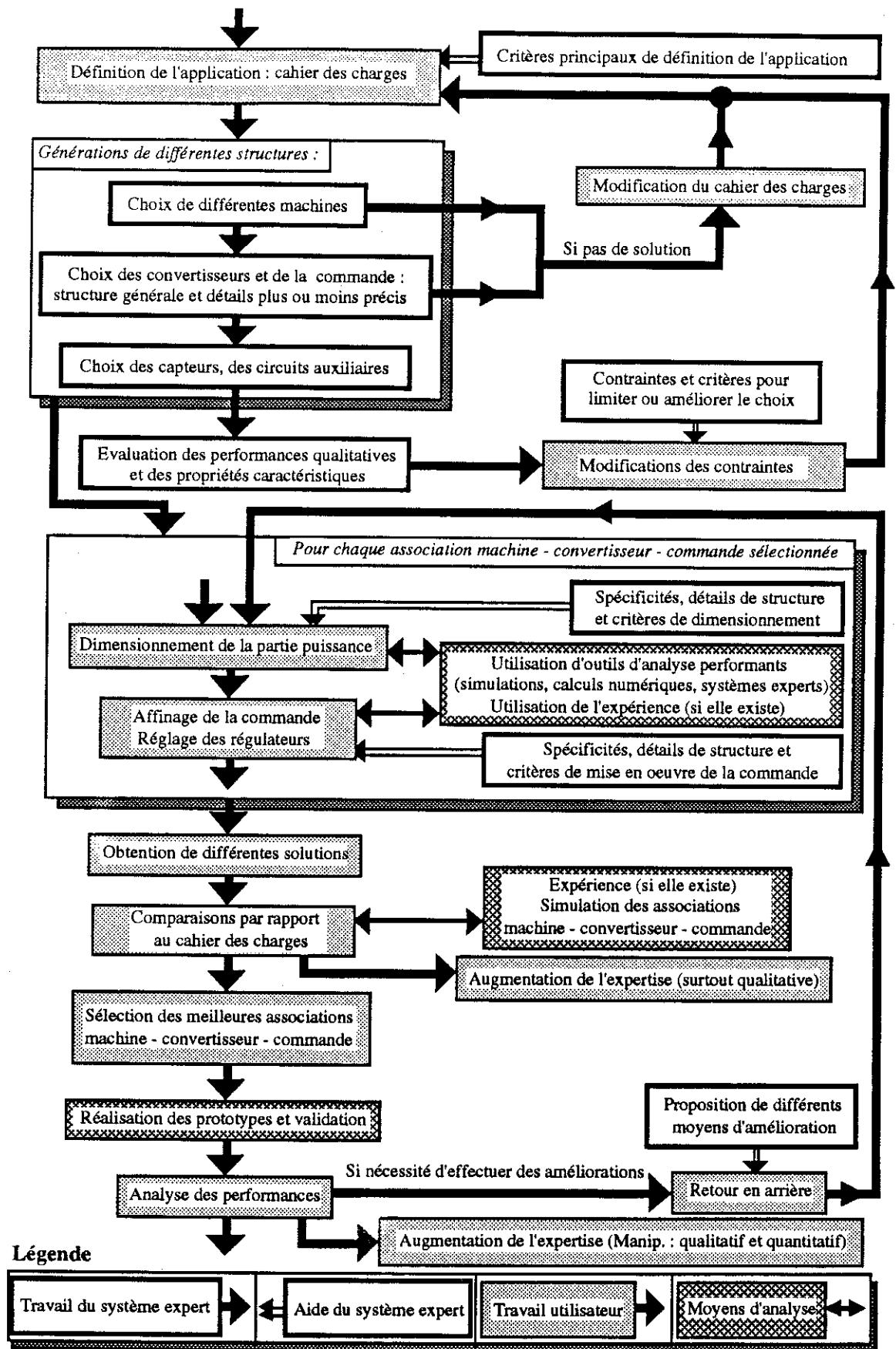


Figure VI.4 : Stratégie de conception proposée

"si condition alors action 1"

"si condition alors action 2"

A partir de ces réalisations, nous avons pu mettre en oeuvre la stratégie de conception dans le système expert.

C. MISE EN OEUVRE DE LA STRATEGIE DE CONCEPTION

La figure VI.4 reprend la stratégie de conception que nous proposons, en distinguant le travail qui est :

- à la charge entière du système expert,
- à la charge entière de l'utilisateur,
- fait par l'utilisateur et assisté par le système expert.

1) REPRESENTATION DE LA STRATEGIE

Nous avons défini une stratégie pour chaque étape du processus de conception, de façon à bien distinguer celles-ci. Le système expert exécute donc la stratégie en donnant un arbre de raisonnement pour chaque étape.

Dans Smeci, l'ordre d'exécution des tâches guidées par les buts, est décrit dans un objet appelé **agenda**. Le moteur d'inférence du système expert exécute ensuite les tâches dans l'ordre donné par cet agenda.

Cependant, certaines tâches sont activées par des règles, lorsqu'elles sont guidées par les données (faits). Informatiquement, cela correspond à l'ajout, ou à la suppression, dynamique de tâches de l'agenda pendant l'exécution du moteur d'inférence.

2) SIMPLIFICATION DE L'ARBRE DE RAISONNEMENT

Nous visualisons l'arbre de raisonnement obtenu pour chaque étape, grâce à l'interface utilisateur de Smeci. Ainsi, l'utilisateur peut avoir accès à toutes les informations contenues dans celui-ci, notamment à la valeur des objets solutions dans chaque contexte terminal.

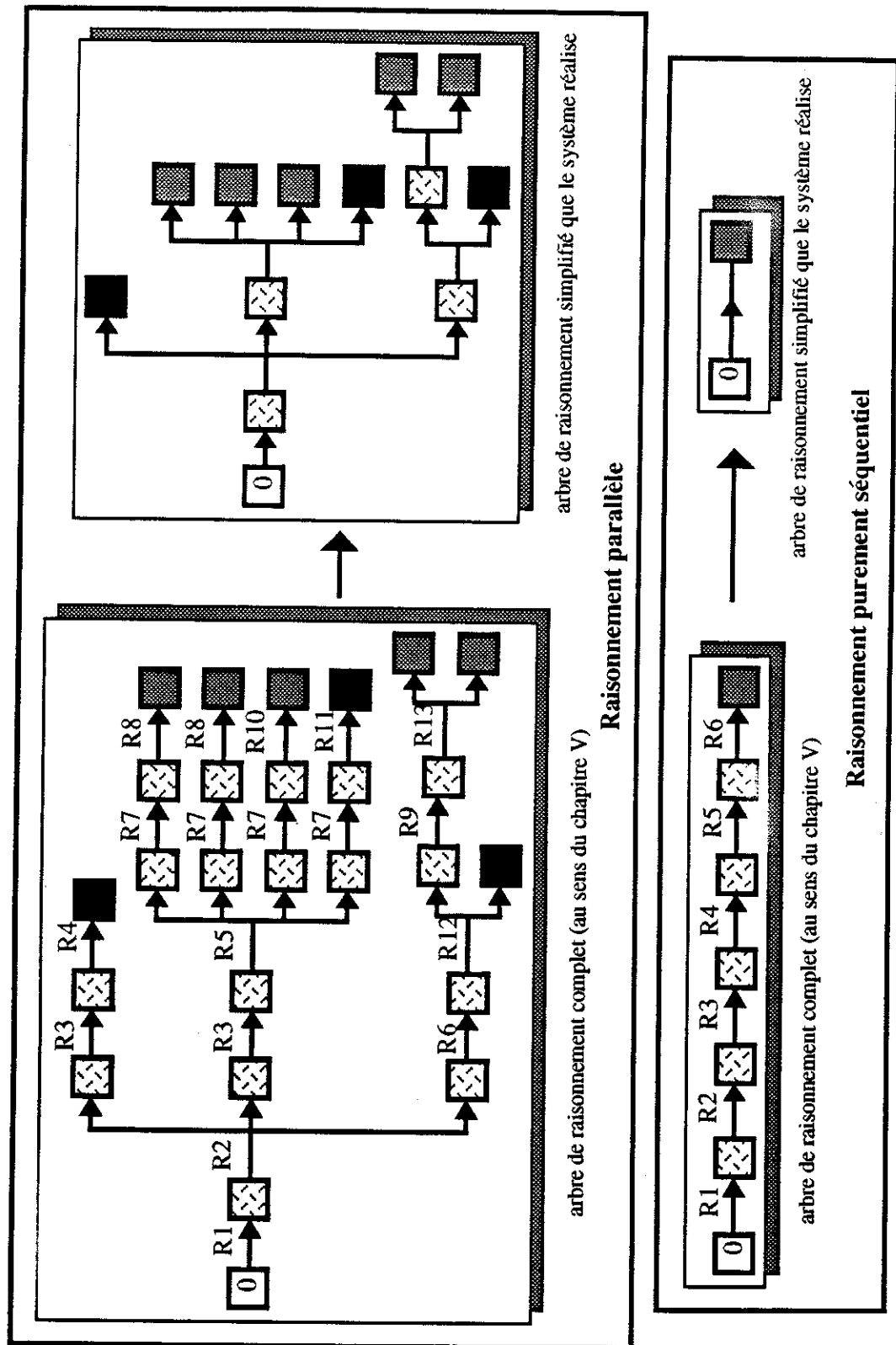


Figure VI.5 : Arbres de raisonnement dans le système d'aide à la conception

Pour simplifier cette visualisation, nous ne faisons apparaître que les éléments importants de ces arbres (figure VI.5) :

- le contexte initial,
- les contextes terminaux,
- les branchements en parallèle.

Remarque : Ce type de simplification n'enlève rien à l'information contenu dans ceux-ci. En particulier, par l'interface du système, l'utilisateur peut connaître le raisonnement (les règles) qui a été appliqué entre deux contextes.

Grâce à tous les éléments présentés en début de ce chapitre, nous avons pu mettre en oeuvre différentes étapes du processus de conception, dans le système expert.

3) MISE EN OEUVRE DU RAISONNEMENT DANS LES ETAPES

La connaissance apparaît dans les objets, dans les règles et dans la stratégie, si bien que notre processus de conception s'exécute suivant trois types d'étape.

a) Etape sans raisonnement

Dans ce type d'étape, nous utilisons des objets dont les valeurs des propriétés et / ou des caractéristiques sont limitées. Ces valeurs sont initialement "vides" mais contraintes ; ainsi, pour chacune d'elles, l'utilisateur doit utiliser uniquement les possibilités prédéfinies qui lui sont proposées (figure VI.6).

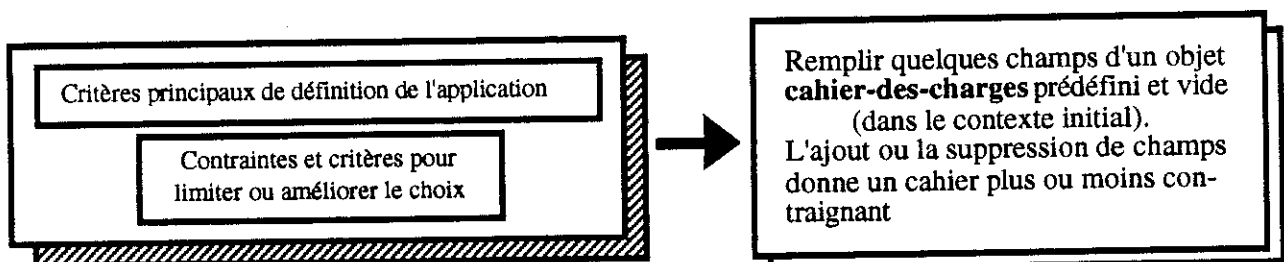


Figure VI.6 : Etape d'élaboration du cahier des charges de l'application à traiter

b) Etapes avec un raisonnement purement séquentiel

Ces étapes donnent un arbre de raisonnement purement séquentiel. Par exemple, cela concerne l'évaluation qualitative des performances des AMCC (figure VI.7).

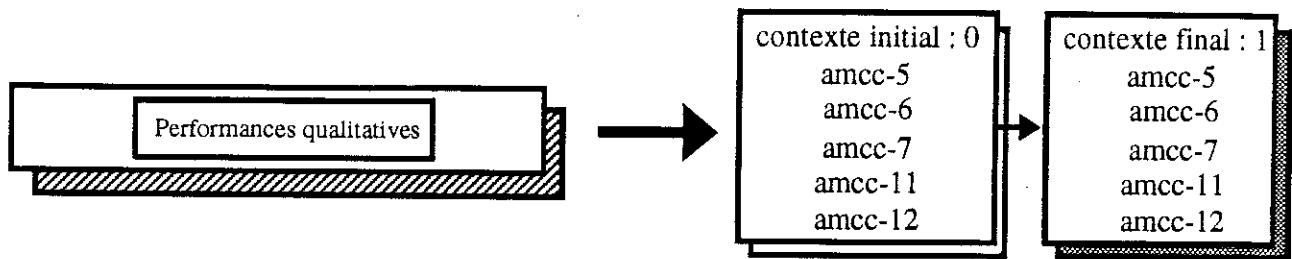


Figure VI.7 : Exemple d'arbre de raisonnement pour les comparaisons qualitatives

c) Etapes avec un raisonnement parallèle

Ces étapes donnent un arbre de raisonnement parallèle. C'est le cas, par exemple, du choix des structures des AMCC répondant au cahier des charges (figure VI.8).

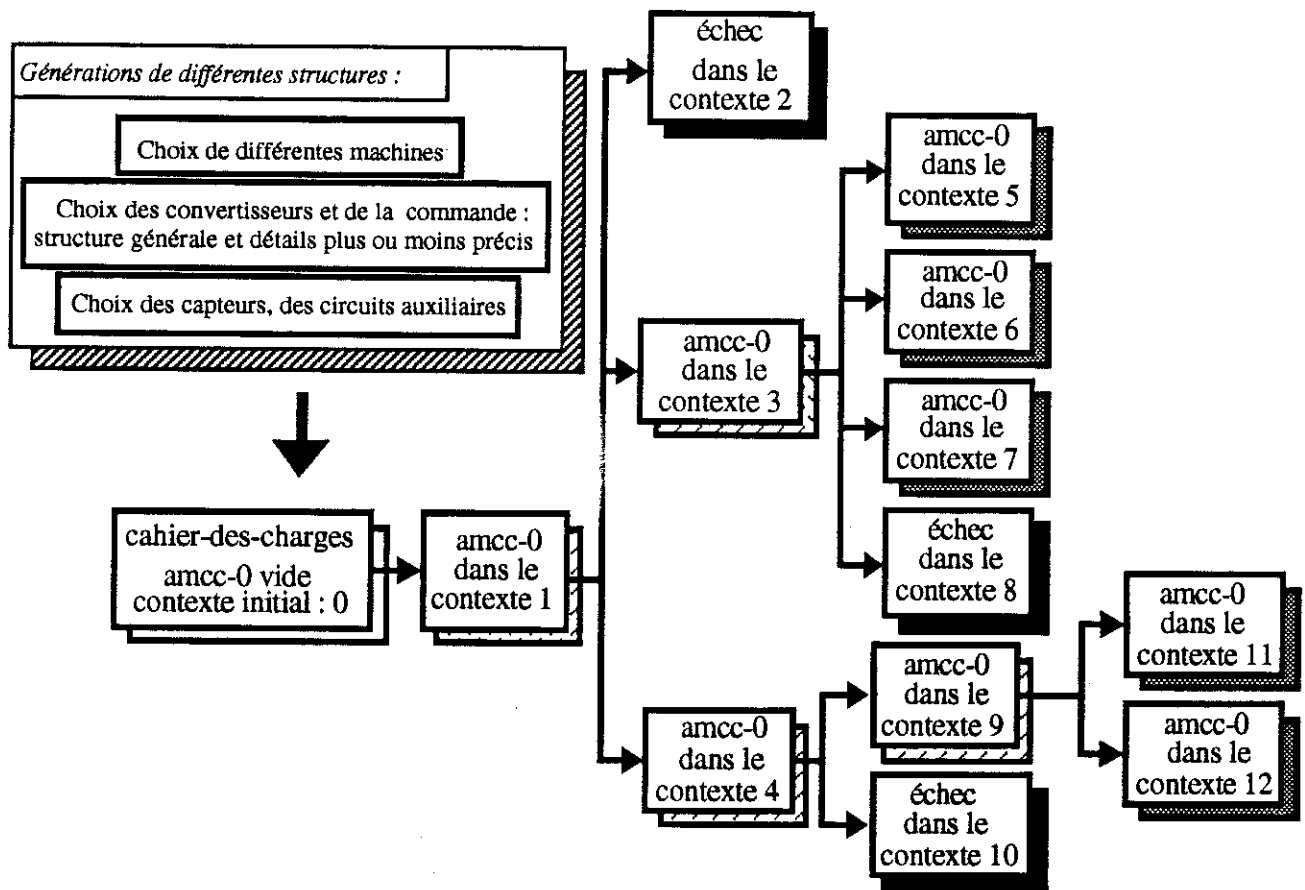


Figure VI.8 : Exemple d'arbre de raisonnement pour le choix des structures des AMCC

Afin de simplifier l'écriture des règles et l'exploitation des résultats obtenus par le système expert, nous avons utilisé, de façon optimale, la notion d'arbre de raisonnement (figure VI.8). Par exemple, dans l'étape choix des structures des AMCC, nous avons réalisé un objet de la classe "AMCC" : amcc-0. Celui-ci est vide dans le contexte initial, et il représente une AMCC

solution dans chaque contexte terminal de l'arbre de raisonnement obtenu par le système. Bien sûr, les valeurs des propriétés et des caractéristiques de cet objet sont différentes d'un contexte à l'autre.

Remarque : Pour faciliter et uniformiser l'accès aux valeurs des caractéristiques et propriétés de l'objet amcc-0 qui sont des objets, nous avons suffixé leur nom par "-0" (figure VI.9).

Objet alimentation-rotor-0		quitter
Sauver Copier Editer () Expliquer Defaire Affecter Inserer Ajouter Supprimer	<pre> prototype = Alimentation name = alimentation-rotor-0 extension = () Convertisseur_cote_source = convertisseur-cote-source-rotor-0 Filtre_alim = filtre-rotor-0 Source_alim = () Harmoniques_alim = () Limitation_de_courant = () Niveau_de_limitation_de_courant = () Circuits_auxiliaires = circuits-auxiliaires-cote-rotor-0 Circuit_dissipatif = circuit-dissipatif-cote-rotor-0 Convertisseur_cote_charge = convertisseur-cote-charge-rotor-0 Alimentation_directe = () Alimentation_directe-1 = () Nom_alim = () Nom_alim-1 = () </pre>	
	alimentation-rotor-0	

Objet amcc-0		quitter
Sauver Copier Editer () Expliquer Defaire Affecter Inserer Ajouter Supprimer	<pre> amcc-0 de Categorie: Amcc prototype = Amcc name = amcc-0 Alimentation_stator = alimentation-stator-0 Commande_amcc = commande-amcc-0 Alimentation_rotor = alimentation-rotor-0 Charge_amcc = Charges-amcc-0 Demarrage_amcc = demarrage-amcc-0 Freinage_amcc = freinage-amcc-0 Comparaisons_alim = comparaisons-alim-0 Comparaisons_commande = comparaisons-commande-0 Comparaisons_amcc = comparaisons-amcc-0 Comparaisons_moteur = comparaisons-moteur-0 Performances_commande = performances-commande-0 Performances_alim = performances-convertisseur-0 Performances_amcc = performances-amcc-0 Performances_moteur = performances-moteur-0 </pre>	
	commande-amcc-0	

Figure VI.9 : Exemples de champs de type objet de la solution amcc-0

4) COMPARAISON DE CONTEXTES

Dans Smeci, les contextes de l'arbre de raisonnement sont complètement indépendants. Ceci signifie qu'il est impossible de les comparer autrement que manuellement.

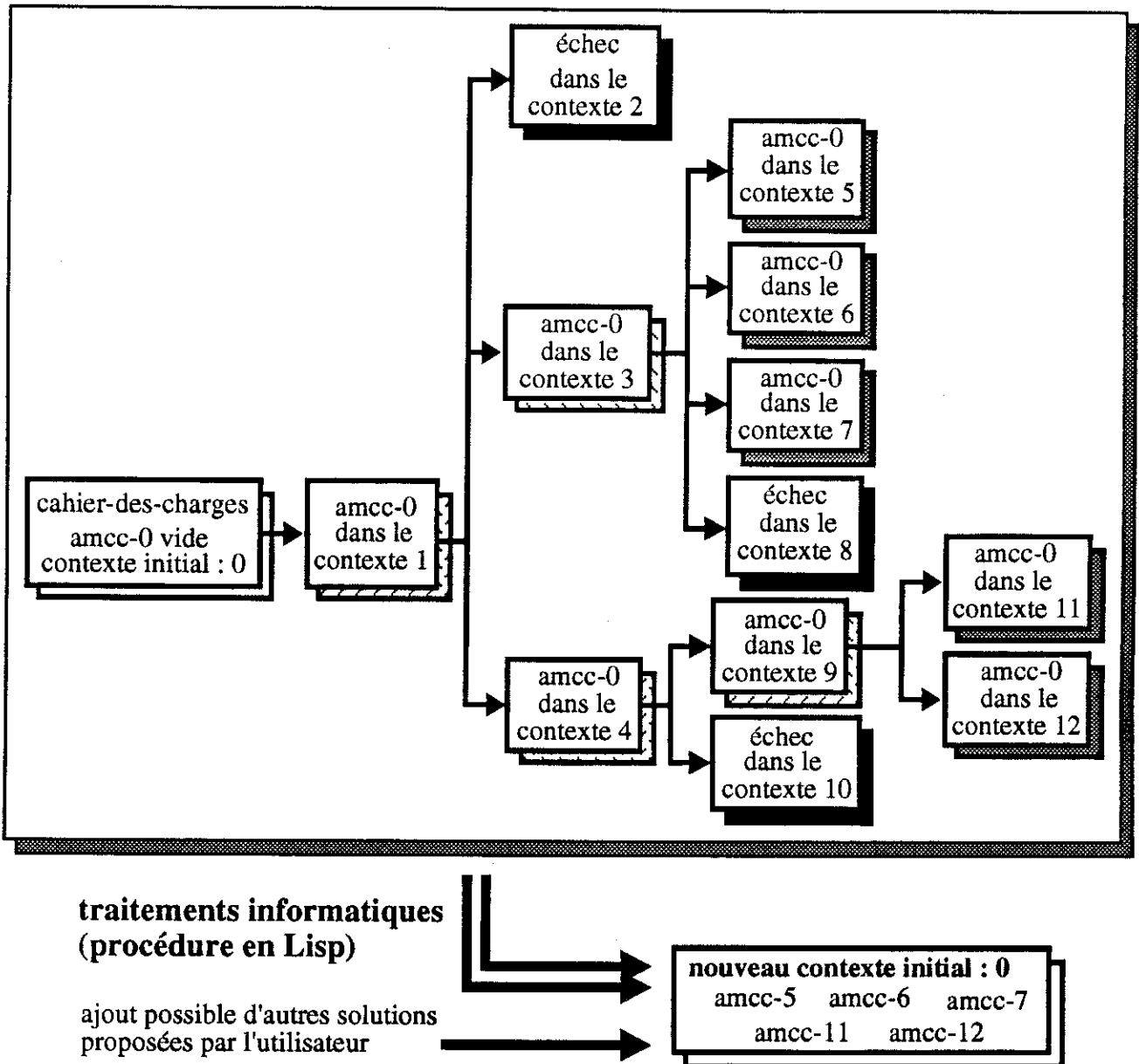


Figure VI.10 : Transformation de n contextes solutions en un seul

Or, nous possédons des règles de comparaison des performances qualitatives des solutions obtenues lors de l'étape "choix des structures des AMCC". Pour pouvoir automatiser l'application de ces règles, nous avons dû élaborer une procédure qui transforme les groupes d'objets suffixés par "-0" dans les n contextes terminaux solutions de l'arbre de l'étape "choix des structures" en n groupes d'objets suffixés par "-i" ($1 \leq i \leq n$) contenus dans le même

contexte (figure VI.10). A partir de ce dernier, toute comparaison qualitative entre les mêmes types d'objets suffixés par "-i" ($1 \leq i \leq n$) est possible (par le système).

5) LE SUIVI DU RAISONNEMENT DE CONCEPTION

Pour aider l'utilisateur à comprendre ce que fait le système, il est important que ce dernier offre un suivi poussé du raisonnement.

Grâce à l'interface, une trace du raisonnement est affichée dynamiquement à l'écran au cours de l'exécution de chaque étape. Ceci concerne les règles appliquées et les tâches en cours d'exécution.

De plus, l'utilisateur dispose de la trace fournie par l'arbre de raisonnement après l'exécution de chaque étape de conception.

6) TRAVAIL EFFECTIVEMENT REALISE

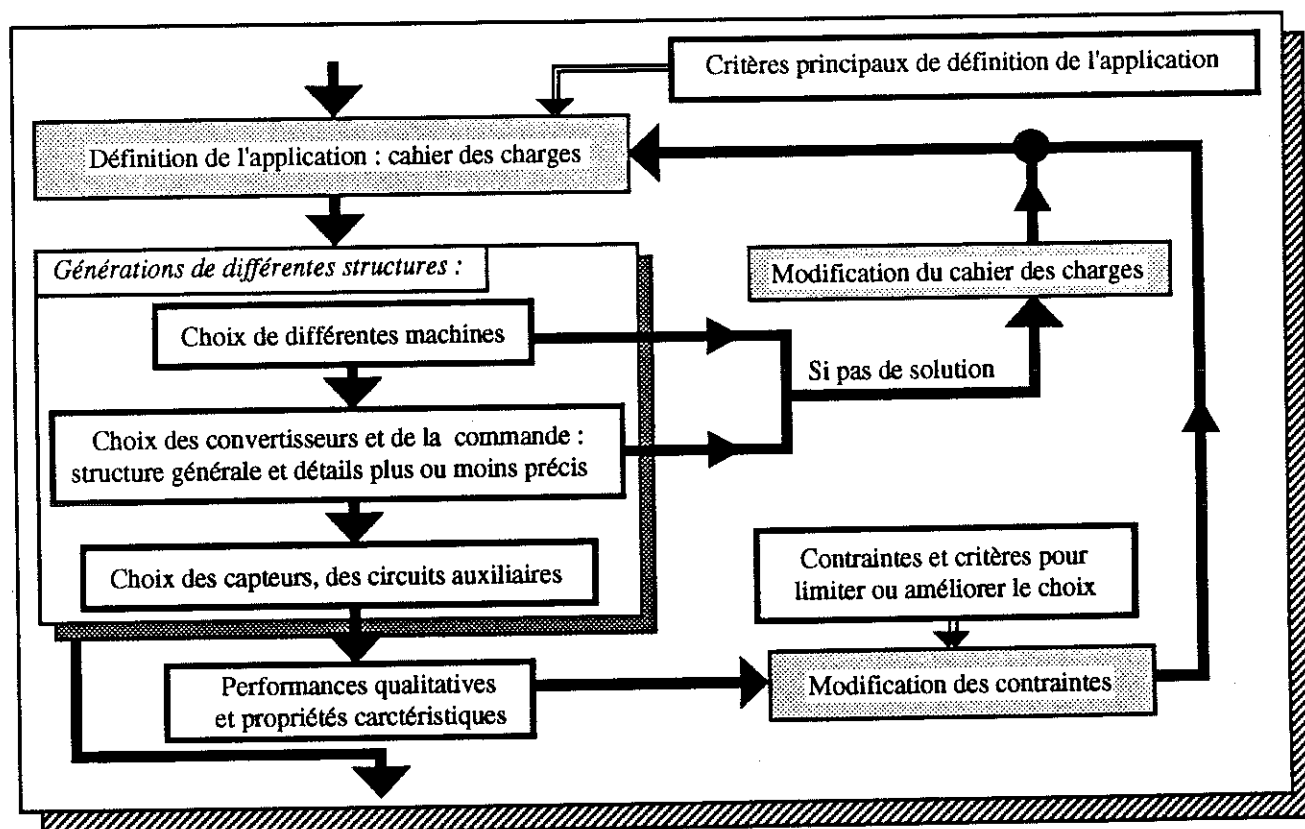


Figure VI.11 : Partie de la stratégie de conception mise en oeuvre informatiquement

Dans le cadre de notre travail, nous n'avons implanté informatiquement et testé que la partie de la stratégie qui est représentée sur la figure VI.11.

Cependant, les autres étapes ont été analysées et testées sur "papier" (à la main). En particulier, tout le travail de reformulation des règles et de leur répartition en tâches a été réalisé. Ces étapes utilisent les considérations évoquées précédemment pour la stratégie (cf § VI.C.1-2-3-4).

Ainsi, pour ces étapes, il ne reste qu'à entrer la connaissance dans le système expert, ce qui est aisé en raison de la généralité du formalisme des règles et des objets.

Nous allons maintenant présenter le déroulement d'une session de conception avec notre système expert. Les points VI.D.1-2-3 concernent le **non spécialiste** qui veut se familiariser avec le processus de conception. La suite concerne plutôt le **concepteur** d'AMCC.

D. UNE SESSION AVEC LE SYSTEME EXPERT

1) ELABORATION DU CAHIER DES CHARGES

L'utilisateur commence par élaborer le cahier des charges de son application. Pour cela, il doit remplir un objet cahier des charges dont les propriétés et les caractéristiques sont "vides". Pour chacune, le système propose des valeurs parmi lesquelles l'utilisateur doit choisir. Il n'est pas obligé de toutes les affecter. De plus, il peut imposer certains éléments de la future AMCC solution, par exemple : la machine (figure VI.12).

Remarque : Notons que dans cette étape, l'utilisateur peut être conduit à effectuer un dimensionnement de sa charge et à réaliser un modèle de celle-ci. De même, il doit dimensionner son application pour connaître la puissance que devra fournir l'AMCC à concevoir.

A partir de là, l'utilisateur active l'étape du "choix des structures" du système expert.

2) GENERATION DES DIFFERENTES STRUCTURES

Dans cette étape, le système commence par choisir les machines qui répondent aux contraintes du cahier des charges, en les éliminant de façon globale. Par exemple, il peut proposer une machine synchrone triphasée à rem trapézoïdale.

Ensuite, pour chaque machine possible, il fournit différentes structures d'alimentation et de commande qui peuvent convenir. Par exemple, il peut indiquer que le convertisseur situé en tête de l'alimentation statorique du moteur (côté source), est un redresseur à diodes (qui transforme

de l'alternatif triphasé en continu monophasé, et qui fonctionne dans un quadrant du repère tension / courant).

Categorie Cahier_des_charges Prototype Cahier_des_charges	
Champs Moteur_existant Alimentation_existante Commande_existante Source_de_tension Cout_equipement_quantitatif Dynamique_commande_vitesse Cout_equipement_qualitatif Cout_maintenance_quantitatif Cout_maintenance_qualitatif Couple_moteur_total_quantitatif Couple_en_bas_regime Couple_moteur_qualitatif Couplage_charge_moteur_impose Puissance_souhaitees Cout_ensemble_quantitatif Cout_ensemble_qualitatif Travail_a_produire_quantitatif Travail_a_produire_qualitatif Temps_de_reponse Temps_de_deplacement_qualitatif Temps_de_deplacement_quantitatif Type_application Vitesse_souhaitees Cout_energie_a_recuperer_quantitatif Cout_energie_recuperer_qualitatif Robustesse Quadrants_de_fonctionnement Freinage_souhaite Recuperation_d_energie Encombrement Type_charge Contrôle_position Environnement_explosif Environnement_humide Taux_de_fiabilite_qualitatif Type_de_commande Contrôle_vitesse Contrôle_couple Contrôle_de_charges_entrainantes Profil_de_vitesse Profil_de_deplacement Fonctionnements_specifiques Rendement Commande_simple Inertie_requise Necessite_blocage	application_vanne application_vanne_fractionnee_en_puissance application_ventilation t () application_a_capacite_variable application_avec_autonomie_electrique application_generatrice appli_peu_de_fonctionnements_a_wr_reduit application_avec_commande_num application_intermittante appli_simu_charge_resistive_sur_rotor Robustesse-1 = Liste de forte faible t () Type_charge-1 = Liste de () charge_centrifuge charge_en_k_fois_wr_puissance_n charge_en_k_fois_wr_au_carre charge_proportionnelle_a_la_vitesse charge_entrainante charge_dt_couple_varie_tres_vite_avec_wr Contrôle_position-1 = Liste de precis t tres_precis () Environnement_explosif-1 = Liste de t () Environnement_humide-1 = Liste de t () Taux_de_fiabilite_qualitatif-1 = Liste de eleve () Contrôle_vitesse-1 = Liste de t () precis tres_precis Contrôle_couple-1 = Liste de precis tres_precis t () Contrôle_de_charges_entrainantes-1 = Liste de
Local Tout Rien	Cahier des charges

Figure VI.12 : Classe (catégorie) et quelques valeurs possibles de champs de l'objet "cahier des charges"

Pour la commande, il peut proposer un contrôle du flux rotorique de la machine, avec deux ou trois boucles de régulation de courant, dans le référentiel tournant dq. Il peut aussi donner la présence d'une limitation du courant machine à deux fois sa valeur nominale.

Le système donne ensuite la liste des capteurs et circuits auxiliaires nécessaires au bon fonctionnement de chaque AMCC solution. Par exemple, il peut imposer la présence d'un circuit dissipatif dans l'alimentation de la machine pour limiter les surtensions, ou pour le freinage de la machine.

Ensuite, le système peut passer à l'étape suivante.

3) EVALUATION QUALITATIVE DES PERFORMANCES ET PROPRIETES

Dans cette étape, le système expert évalue qualitativement les performances et les propriétés des différentes structures obtenues, et effectue des comparaisons qualitatives entre celles-ci. Par exemple, il indique que la machine à courant continu d'une structure est 3 à 7 fois plus chère que la machine asynchrone d'une autre structure. De même, il précise s'il y apparaît des harmoniques de courant en 6ω ou 12ω (ω : pulsation statorique).

La figure VI.13 donne quelques exemples de ces performances (ou comparaisons).

Chacune est caractérisée par :

- Un nombre qui représente une **note** affectée par le système. Nous l'avons choisi positive si l'aspect évoqué est avantageux, négative sinon. Par exemple, un coût élevé par rapport à d'autres solutions donne une note négative, et une complexité de commande positive correspond à une commande simple (c'est un aspect positif !).
- Une **liste d'explications** de la note qui donne une trace des règles ayant affecté ou modifié cette note. Ainsi, l'utilisateur peut savoir par rapport à qu'elle autre AMCC, une AMCC a été comparée.

Il y a environ une cinquantaine de critères qui peuvent être affectés par le système. Grâce à ceux-ci, l'utilisateur peut se faire une idée des quelques performances qui l'intéressent et des solutions qu'il doit privilégier dans la suite du processus de conception.

A partir de cette évaluation, l'utilisateur peut décider de remodifier les contraintes du cahier des charges pour obtenir une solution qui soit qualitativement plus satisfaisante.

Categorie Comparaisons

Champs

Rendement
 Explication_rendement
 Cout
 Explication_cout
 Cout_achat
 Explication_cout_achat
 Cout_maintenance
 Explication_cout_maintenance
 Stabilite
 Explication_stabilite
 Facteur_de_puissance
 Explication_facteur_de_puissan
 Flexibilite
 Explication_flexibilite
 Competitivite
 Explication_competitivite
 Preference
 Explication_preference
 Complexite
 Explication_complexite
 Pertes
 Explication_pertes
 Poids
 Explication_poids
 Taille
 Explication_taille
 Aspects_identiques
 Performances
 Explication_performances
 Aspects_differeents
 Aspects_particuliers
 Temps_de_reponse
 Explication_temps_de_reponse

Categorie Comparaisons_amcc

Champs

Precision_capteur_position
 Explication_precision_capt_pos
 Cout_capteur
 Explication_cout_capteur
 Cout_electronique
 Explication_cout_electronique
 Precision_capteur_vitesse
 Explication_precision_capt_vit
 Domaine_d_emploi
 Explication_domaine_d_emploi
 Performances_tension
 Performances_courant
 Performances_fem
 Performances_couple
 Performances_vitesse
 Pulsation_de_couple
 Explication_pulsation_de_couple
 Chauffage_moteur
 Explication_chauffage_moteur
 Gamme_de_puissance
 Gamme_de_vitesse
 Declassement_moteur
 Explication_declassement_m

Categorie Comparaisons_convertisseur

Champs

Possibilite_multimoteur
 Explication_multimoteur
 Protection_circuit_ouvert
 Explication_protection_co
 Protection_court_circuit
 Explication_protection_cc
 Commande_de_mot_ss_dim
 Explication_com_de_mot_ss_dim
 Commande_de_mot_sur_dim
 Explication_com_de_mot_sur_dim
 Possibilite_fonct_direct
 Explication_fonct_direct
 Fonct_generateur
 Explication_fonct_generateur

Figure VI.13 : Exemples de performances ou comparaisons

Remarque : Pour obtenir une évaluation globale des performances, il est possible de réaliser une fonction pondérée. Dans celle-ci, chaque note peut être prise ou non en compte, avec un coefficient permettant d'en privilégier certaines par rapport à d'autres. Ceci permettrait, par exemple, de donner plus d'importance aux pertes qu'au poids de l'AMCC solution. Actuellement, ceci reste à la charge entière de l'utilisateur, mais on peut facilement envisager de réaliser une telle fonction dans le système.

Après avoir sélectionné les meilleures solutions (qualitatives), l'utilisateur peut effectuer leur dimensionnement.

4) DETAILS DE STRUCTURE ET DIMENSIONNEMENT

Ces étapes ne sont pas automatisées. Toutefois, des règles ont été répertoriées et structurées ; et elles sont disponibles pour être implantées dans le système expert.

Avec celles-ci, le système fournira des détails de conception pour les différents éléments des AMCC solutions. Il pourra fournir des critères de dimensionnement pour prendre en compte les interactions mutuelles entre les différents composants des AMCC. Par exemple, il fournira un niveau limite du courant traversant les interrupteurs des convertisseurs ou il déclassera la machine en puissance.

Pour la commande, il fournira essentiellement les principes et lois de commande, ainsi que les propriétés de la machine qui sont utilisées. Pour les détails des structures des boucles de contrôle, il est plus succinct et invite le concepteur à consulter des ouvrages pour plus de précision.

Après avoir défini avec précision la, ou les, AMCC qu'il souhaite concevoir, le concepteur peut consulter des catalogues de fabricants, ou il peut envisager de concevoir lui-même ces composants. Ce dernier point peut être réalisé en utilisant des systèmes experts d'aide à la conception, pour, par exemple concevoir des machines asynchrones à cage [TRI91] ou des convertisseurs statiques [AHR90] [CHH92].

Remarque : Dans ces étapes, le travail réalisé par le concepteur (ou l'utilisateur) est important ; le système expert ne joue qu'un rôle de conseil et d'aide.

5) ANALYSE DES PERFORMANCES

Dans cette étape, l'utilisateur effectue une première analyse qualitative et quantitative des

solutions qu'il vient de dimensionner avant de passer au prototype. Pour cette étude, il s'aide d'outils de simulation (dont celui développé au début de cette thèse) et de l'expérience, si elle existe.

A partir de là, il choisit les meilleures solutions pour lesquelles un prototype sera réalisé. Puis il commence à enrichir le système expert de son analyse.

6) PROTOTYPE ET ETUDE DES SOLUTIONS

Avec les prototypes, des études plus approfondies peuvent être réalisées. Dans cette étape, il est possible que l'utilisateur souhaite améliorer son AMCC. Pour l'aider, le système expert lui fournit une aide sous forme de conseils (règles de bon sens et de savoir faire).

Cette phase d'amélioration, voire d'optimisation des solutions, peut reconduire l'utilisateur à redimensionner son prototype ou les boucles de régulation, donc à reprendre la phase présentée au paragraphe D.4.

Là encore, une expertise peut être extraite pour enrichir le système expert.

Nous présentons maintenant quelques résultats issus du système expert.

E. RESULTATS

La stratégie implantée dans le système expert a été testée sur de nombreux exemples dont le cahier des charges était plus ou moins général.

Nous avons trouvé des solutions correctes pour chaque cas, mais parfois elles étaient incomplètes en raison du manque d'expertise contenue dans le système expert.

Par exemple, pour le cahier des charges suivant :

- source de tension : réseau triphasé,
- application de levage,
- réversibilité dans les quatre quadrants du repère couple-vitesse.

notre système donne une seule solution constituée d'une machine à courant continu dont

l'alimentation rotorique contient :

- un redresseur triphasé à diodes (transformant du triphasé en monophasé),
- un filtre de tension,
- un circuit dissipatif qui sert pour un freinage dissipatif et pour limiter les surtensions,
- un hacheur quatre quadrants commandé en simple modulation de largeur d'impulsion,
- une limitation du courant d'induit pendant le démarrage ou les montées en vitesse.

Cette solution est caractérisée par les performances et spécificités suivantes :

- hacheur réversible dans quatre quadrants (courant-tension),
- oscillations du courant dans le moteur,
- oscillations crête à crête du courant rotor : constantes,
- échauffement du rotor car oscillations de courant,
- pulsations de couple car oscillations du courant rotor.

Remarque : C'est le critère "application de levage" qui a imposé la machine, car un expert conseillait fortement celle-ci pour une telle application.

Avec un cahier des charges plus général :

- source de tension : réseau triphasé,
- contrôle de vitesse précis;

notre système donne beaucoup plus de solutions et effectue des comparaisons qualitatives de celles-ci. Nous ne présentons que l'arbre de raisonnement de l'étape "choix des structures" où nous explicitons les différences essentielles entre les dix AMCC obtenues (figure IV.14). Le système fournit beaucoup d'autres informations, notamment sur les circuits d'aide à la commutation des convertisseurs.

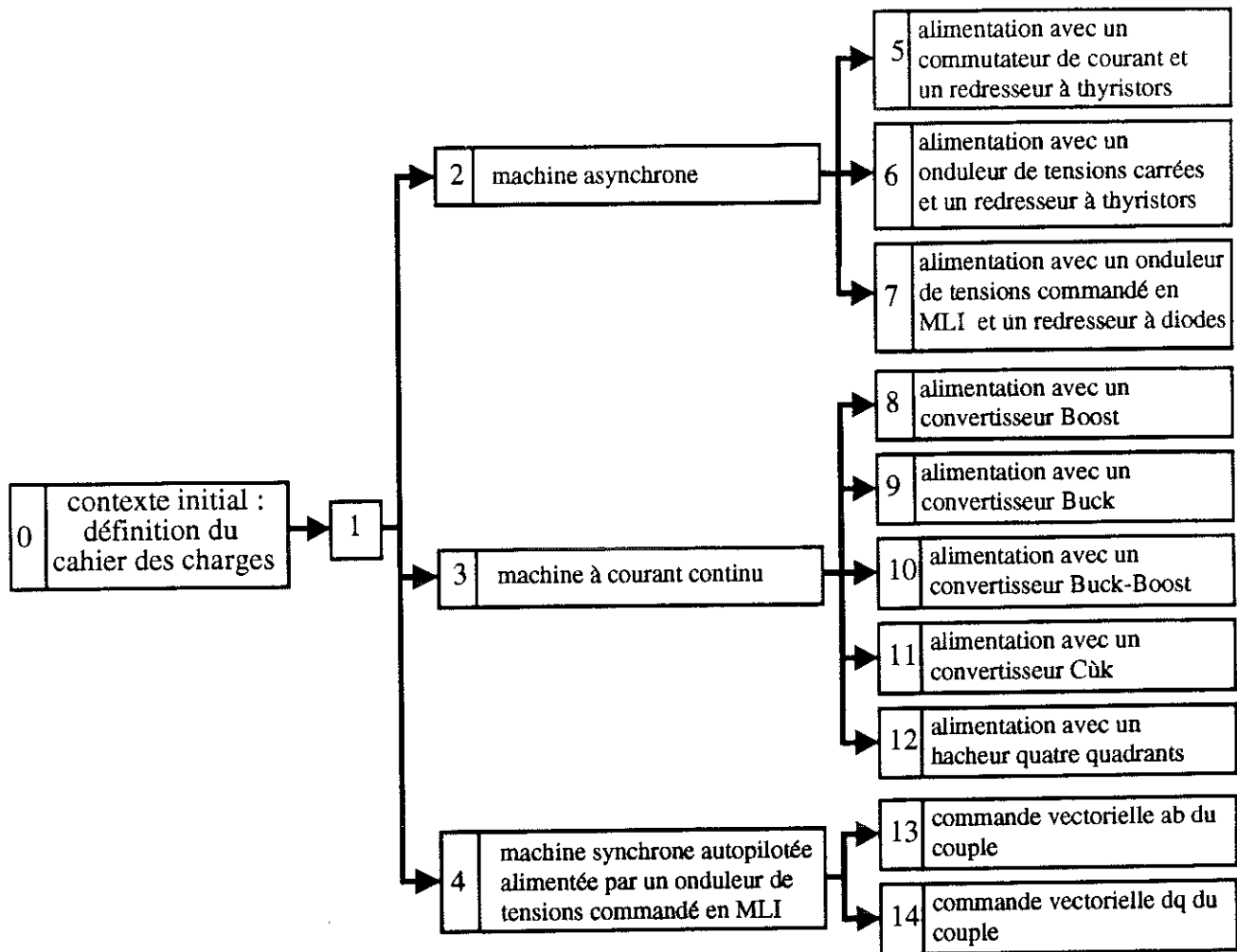


Figure VI.14 : Résultats principaux obtenus pour le choix des structures

Remarque : Si l'application avait été de très forte puissance et de faible vitesse, le système expert aurait proposé une structure avec une machine asynchrone alimentée par un cyclo-convertisseur.

F. APPORTS DU SYSTEME EXPERT

Ce système d'aide à la conception offre plusieurs avantages :

- Il permet de répertorier facilement toute une expertise sur la conception des AMCC (règles, faits, stratégie).
- Il propose une stratégie de conception dont il réalise automatiquement certaines étapes, et pour laquelle il propose un aide au concepteur d'AMCC (par des règles de savoir-faire).

- Il permet au non spécialiste ainsi qu'au client du concepteur d'AMCC, de se familiariser avec le processus de conception. En particulier, il aide le client à définir le cahier des charges de son application.
- Le nombre de contraintes que doit préciser l'utilisateur dans le cahier des charges n'est pas imposé. De plus, il peut contraindre le système à chercher une AMCC où la machine est définie dans le cahier des charges.
- Le concepteur peut utiliser le système pour connaître les performances qualitatives d'une AMCC en activant uniquement l'étape du processus correspondante.

Ce système introduit l'utilisation de l'outil de simulation présenté au début de cette thèse avec tous les apports et avantages issus de l'utilisation du calcul formel (cf chapitre II.C).

Il constitue donc les bases solides d'un système d'aide à la conception d'AMCC que l'on pourra améliorer.

Conclusion

CONCLUSION

Les travaux réalisés dans le cadre de cette thèse nous ont permis de proposer une aide à la conception d'ensembles machine électrique - convertisseur - commande, grâce à l'utilisation de logiciels de simulation associés à un système expert.

L'expérience acquise en simulation numérique de ces ensembles autour du logiciel MC2, nous a permis de définir les exigences et les critères principaux d'élaboration d'un tel outil : minimiser le travail d'un utilisateur désirant décrire une nouvelle application et assurer une précision et une fiabilité maximales des résultats de simulation. Pour cela, nous avons été conduits à mettre en oeuvre un outil assurant une résolution globale des équations et générant automatiquement les programmes de simulation correspondants.

Cet outil nécessite une bibliothèque de modèles de composants (machines, convertisseurs statiques ...) pour le rendre facilement utilisable. Actuellement, la création de ces modèles (blocs) est "manuelle". Ainsi, pour les convertisseurs statiques qui sont les plus longs à décrire, on peut envisager de développer un outil réalisant automatiquement leur modèle (génération des configurations et de leurs tests d'enchaînement).

L'outil de simulation, présenté dans cette thèse, offre l'avantage de pouvoir être couplé avec un système expert d'aide à la conception pour l'évaluation des performances des associations obtenues. Dans ce cas, le système expert donnera la liste des composants utilisés et la façon dont ils sont agencés dans chaque association sélectionnée, puis l'outil de simulation générera le logiciel exécutable correspondant.

Pour réaliser un système expert d'aide à la conception d'associations machine - convertisseur - commande, nous avons dû formaliser l'ensemble de la démarche de conception et faire en sorte que celle-ci puisse évoluer.

Pour montrer les avantages de l'emploi d'un tel système, nous avons entièrement mis en oeuvre la première phase de la démarche (choix des structures répondant au cahier des charges et évaluation qualitative de leurs performances).

Dans un premier temps, nous avons dû organiser et structurer la connaissance que nous avons recueillie, pour assurer le bon fonctionnement du système. Nous avons veillé à ce que ses bases de connaissances puissent facilement être modifiées et complétées par tout expert qui validera le système. Cette expertise peut être issue de l'analyse réalisée lors de la conception d'une nouvelle association.

Cependant, afin de rendre le système plus performant, nous avons constaté qu'il faudra affiner les modèles utilisés pour représenter l'imprécis et l'incertain contenus dans l'expertise. Pour cela, des techniques issues de la logique floue pourront être utilisées.

Dans un second temps, nous envisageons de mettre en oeuvre une pondération de certains critères du cahier des charges (coût, performances ...), en donnant des priorités (affectées par l'utilisateur) pour chacun d'eux, dans le processus de choix des structures des associations possibles.

L'analyse des autres phases de la conception (dimensionnement, analyse, optimisation, validation expérimentale ...) devra être approfondie. Ceci représente un développement important de recherche. Par exemple, l'optimisation du dimensionnement des différents composants nécessitera le couplage de plusieurs outils (systèmes experts de conception de machines et de leurs alimentations (convertisseurs), et outil de simulation tel que celui présenté dans ce mémoire...).

Annexes

ANNEXE I

MC2 [BOL89]

Vers 1988, après différentes expériences en simulation d'AMCC (Associations Machine - Convertisseur - Commande), en vue de l'étude de la commande, un outil général pour ce type de simulation a été créé au sein du L.E.G. Cet outil, appelé MC2 (Machine-Convertisseur-Commande), propose de structurer le travail de programmation à faire pour obtenir un logiciel de simulation d'AMCC.

Il offre un cadre de programmation de modèles et il libère l'utilisateur de tous les aspects purement informatiques d'un outil de simulation et de ses interfaces :

- méthodes de résolution numérique de systèmes différentiels,
- gestion des résultats : tracé de courbes, sauvegarde,
- sauvegarde des problèmes,
- modification et visualisation interactives des données,
- choix entre différents modèles déjà programmés.

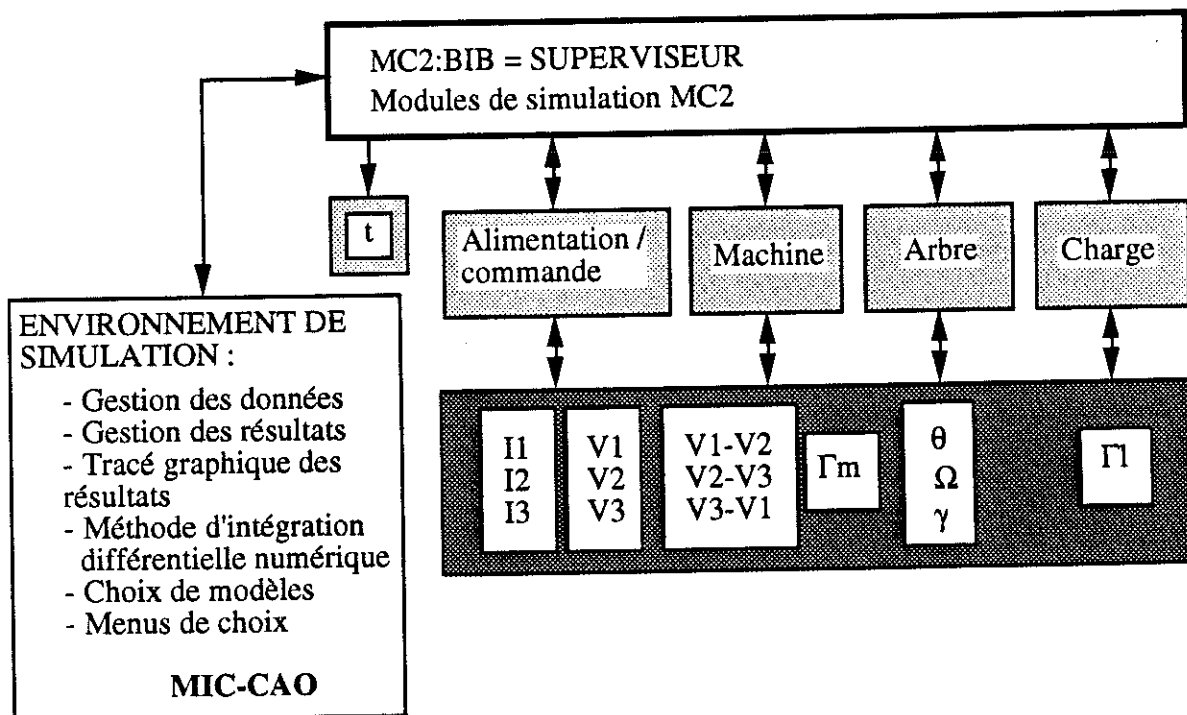


Figure 1 : MC2

MC2 apparaît à la fois comme un outil de recherche et un outil pédagogique permettant de comparer différents modèles, différentes commandes ...

Il propose une décomposition modulaire des AMCC, en quatre modules (cf figure 1):

- Alimentation / commande,
- Machine,
- Arbre,
- Charge.

qui ont en commun quinze grandeurs

DUMACC(1)	: V_1-V_2 : tension entre lignes 1 et 2, à l'entrée de la machine,
DUMACC(2)	: V_2-V_3 : tension entre lignes 2 et 3, à l'entrée de la machine,
DUMACC(3)	: V_3-V_1 : tension entre lignes 3 et 1, à l'entrée de la machine,
DVCOMC(1)	: V_1 : tension phase 1-neutre à la sortie de l'alimentation,
DVCOMC(2)	: V_2 : tension phase 2-neutre à la sortie de l'alimentation,
DVCOMC(3)	: V_3 : tension phase 3-neutre à la sortie de l'alimentation,
DIMACC(1)	: I_1 : courant dans la phase 1 de la machine,
DIMACC(2)	: I_2 : courant dans la phase 2 de la machine,
DIMACC(3)	: I_3 : courant dans la phase 3 de la machine,
DGAMC	: Γ_m : couple moteur,
DGACHC	: Γ_{ch} : couple de charge,
DACCEC	: γ_r : accélération angulaire du rotor,
DVITRC	: Ω_r : vitesse angulaire du rotor,
DPOSRC	: θ_r : position angulaire du rotor,
DTC	: t : temps courant.

Chacun de ces modules peut être décomposé en blocs. Leur résolution se fait séquentiellement, dans l'ordre suivant : Alimentation / commande, machine, charge, arbre.

A chaque pas de calcul, selon la forme sous laquelle sont réorganisées les équations de puissance et de commande analogique (cf chapitre II), la résolution dynamique des équations se fait suivant l'un des deux organigrammes de la figure 2.

Enfin, il faut noter que MC2 répond aux niveaux (a) et (b) de service d'un outil de simulation (§ I.B.1).

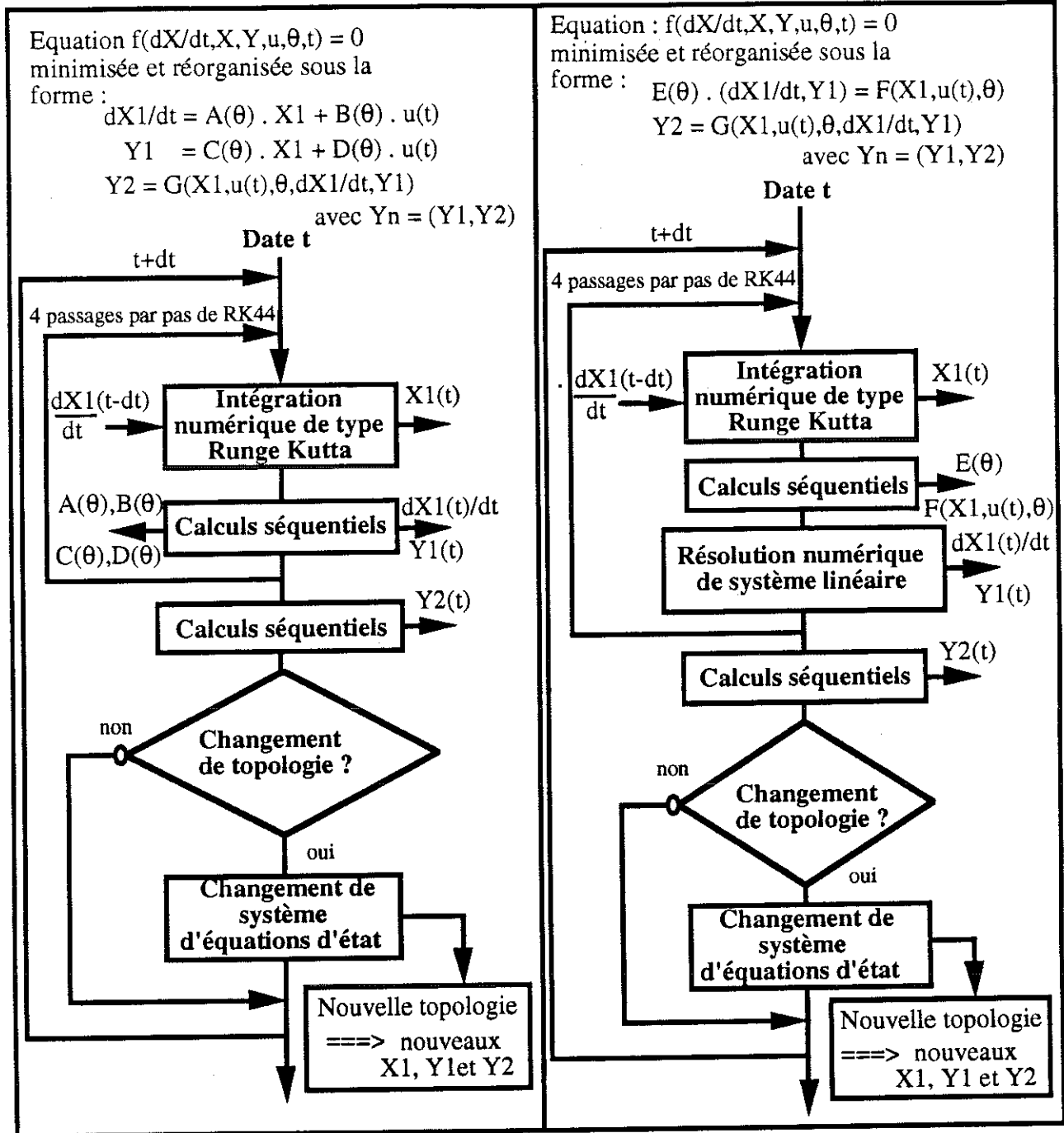


Figure 2. : Organigrammes de résolution des équations d'état

ANNEXE II

MACSYMA [MAC88]

1) PRESENTATION GENERALE

Le calcul formel (ou symbolique), permet de faire de nombreuses manipulations mathématiques et analytiques, qui sont parfois très complexes manuellement.

Par rapport aux calculs numériques traditionnels, il a l'avantage de fournir une solution exacte. En effet, une approche numérique présente l'inconvénient de fortement dépendre de la représentation choisie pour les réels et de la façon dont les opérations de base, en particulier arithmétiques, sont effectuées.

Parmi les différents langages de calcul formel, Macsyma est l'un des plus anciens. Il a été développé à partir de 1969, au Massachusetts Institute of Technology. La version 13, dont nous disposons, est en COMMON LISP et correspond au release 415.25. Elle fonctionne sur PC 80386, sous MS-DOS3 ou plus, et sous WINDOWS. D'autres versions sont implantées sur d'autres systèmes d'exploitation, et sur des stations de travail. Il a été conçu en même temps que Reduce qui est un système développé par A.C. Hearn pour l'électrodynamique quantique.

Actuellement, Macsyma commence à être de conception ancienne. Il est fortement concurrencé par des systèmes tels que MAPLE, Mathematica, Scratchpad II [PIN].

Maple est né en 1980, d'une réflexion sur l'utilisation du calcul formel pour l'enseignement, à l'Université de Waterloo, au Canada. Il est très accessible et pédagogique.

Mathématique, annoncé vers 1988, est un système de calcul formel réalisé par une petite équipe, autour de S. Wolfram aux Etats-Unis. Il est surtout caractérisé par sa convivialité, sa revue et le marketing important qui est fait autour.

Scratchpad II est un système développé au Centre de Recherche T. Watson d'IBM, à Yorktown Height, près de New-York et écrit en 1977-1978. C'est le seul système de calcul formel typé. Il est caractérisé par une approche objet très structurée et la possibilité de compiler des fonctions. Cependant, il est encore à l'état de prototype et il est mis uniquement à la disposition des universitaires.

2) FONCTIONNALITES DE MACSYMA

Macsyma est un des systèmes de calcul formel les plus complets. Afin de donner un aperçu (non exhaustif) de ses possibilités, les principales fonctionnalités de Macsyma sont listées dans ce paragraphe.

En arithmétique et algèbre, Macsyma permet :

- la manipulation de nombres (rationnels, complexes, constantes π , exponentielle ...),
- la création et l'utilisation de fonctions (Bessel ...),
- les manipulations matricielles (produits, déterminants, valeurs propres ...),
- la résolution d'équations,
- des simplifications algébriques et trigonométriques,
- des extractions de termes ou des substitutions d'éléments, dans des expressions mathématiques,
- des sommes et produits de séries,
- des manipulations d'ensembles (union, intersection ...).

En calcul, Macsyma permet :

- les calculs différentiels et les calculs de limite,
- les intégrations,
- la résolution d'équations différentielles classiques,
- les transformées de Laplace et Fourier,
- l'analyse vectorielle.

Macsyma permet aussi :

- la génération de code FORTRAN ou C,
- le tracé de fonctions 2D ou 3D.

Enfin, son utilisateur dispose d'une aide en ligne (interactive) par menus.

Nous avons utilisé Macsyma en tant que langage de programmation de haut niveau pour :

- faire des boucles et des fonctions de programmation,
- traiter des variables symboliques,
- réorganiser des équations,

- générer des expressions et du code FORTRAN,
- simplifier des formules.

3) INCONVENIENTS ET REPROCHES

Pour notre application, les principales remarques sur Macsyma sont :

- Dans notre version, il n'est accessible que par WINDOWS. Ainsi, il ne peut être lancé en batch, contrairement à une version UNIX.
- Dans notre cas, la gestion des fichiers est très délicate et elle a nécessité l'utilisation du Lisp.
- Dans un fichier de commande Macsyma, il est impossible d'accéder au Lisp de Macsyma.
- Contrairement aux autres versions, Macsyma PC ne permet pas l'appel de fonctions FORTRAN.
- Macsyma PC-DOS ne permet pas les accès au système d'exploitation de l'ordinateur.
- Le calcul formel peut donner des formules très lourdes qu'il faut retraiter, simplifier et qui peuvent conduire à une résolution numérique finale moins efficace qu'une démarche totalement numérique. Notons que cette remarque est générale et s'applique à tous les systèmes de calcul formel.

Enfin, Macsyma est écrit en Lisp, mais il ne donne pas accès à son code source. Il est donc délicat de connaître la représentation interne des objets manipulés et le détail exact des algorithmes implantés. Les fonctions écrites par l'utilisateur restent interprétées, ce qui est un facteur important de lenteur d'exécution. Si l'utilisateur souhaite compiler ses fonctions, il doit les écrire en Lisp et assurer l'interface avec Macsyma. D'après l'avis d'experts, ce travail, qui n'est pas de la plus grande facilité, reste réservé aux spécialistes.

Notons que dans notre application, la lenteur de Macsyma n'est pas trop problématique ; par contre, l'impossibilité de compiler les fichiers de commande, empêche toute protection des fichiers écrits en Macsyma de notre générateur.

ANNEXE III

ENTREE DES DONNEES D'UN PROBLEME AVEC NOTRE GENERATEUR

1) SYMETRIES

Afin de faciliter l'entrée des données et de limiter le nombre de jeux d'équations (c'est-à-dire de configurations), le générateur utilise toutes les symétries de l'AMCC étudiée. Par exemple, pour un redresseur triphasé à thyristors ou à diodes (cf figure 1), sept configurations peuvent être considérées en fonctionnement normal. Ceci est possible si on représente le réseau par un modèle idéal, c'est-à-dire sans modéliser les effets capacitifs et inductifs. Ainsi, le redresseur est représenté sans phase d'empiètement d'anode.

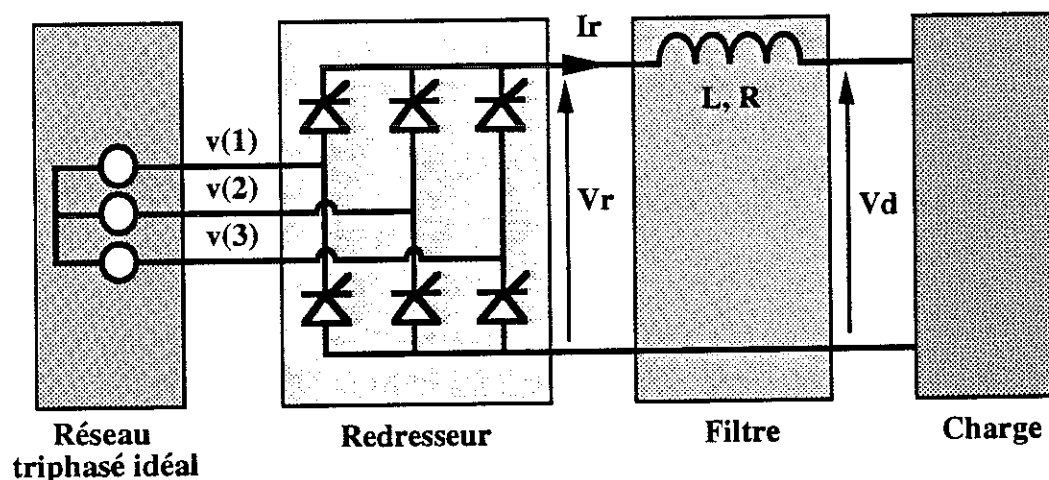


Figure 1 : Modèle du redresseur triphasé

Cependant, en raison des symétries des trois phases, si on indice les tensions du réseau, uniquement deux jeux d'équations sont nécessaires pour décrire son fonctionnement, soit :

- système (1) : $I_r = 0$ configuration (1)
- système (2) : $V_r = v(\text{indice}(1)) - v(\text{indice}(2))$

Ce dernier système représente les configurations 2 à 7, en utilisant une permutation cyclique de la valeur de indice (1) et de indice (2), selon le tableau 2. Indice (1) et indice (2) sont affectés en cours de simulation numérique, par le sous-programme gérant les configurations du redresseur. Pour notre générateur, il faut seulement déclarer l'équation $V_r = v(1) - v(2)$ et fournir les

informations qui lui indiqueront de faire une indexation avec le vecteur indice, sur la variable v qui est de dimension 3.

N° de configuration	Indice(1)	Indice(2)	Indice(3)
1	1	2	3
2	1	3	2
3	2	1	3
4	2	3	1
5	3	1	2
6	3	2	1

Tableau 2 : Permutation cyclique d'indice d'ordre 6

Si les trois phases du réseau ne sont pas symétriques, il est alors difficile d'indicer cette équation. Dans ce cas, à partir de l'équation $V_r = v(1) - v(2)$ et d'une information indiquant de faire une permutation cyclique d'indice d'ordre 6 sur la variable v , notre générateur générera les six équations pour les configurations 2 à 7 qui sont présentées sur le tableau 3.

N° de configuration	équation générée automatiquement
i	$V_r = V(1) - V(2)$
$i + 1$	$V_r = V(1) - V(3)$
$i + 2$	$V_r = V(2) - V(1)$
$i + 3$	$V_r = V(2) - V(3)$
$i + 4$	$V_r = V(3) - V(1)$
$i + 5$	$V_r = V(3) - V(2)$

Tableau 3 : Exemple de permutation cyclique d'ordre 6

Les jeux d'équations générés sont numérotés par l'indicateur de configuration, qui varie de i (donné par l'utilisateur) à $(i + 5)$. Cet indicateur permet d'établir le lien entre ces jeux d'équations et le programme qui gère leur enchaînement en cours de simulation.

Des considérations semblables permettent de réaliser des permutations cycliques d'indices d'ordre 3 (figure 4) et d'ordre 2 (figure 5).

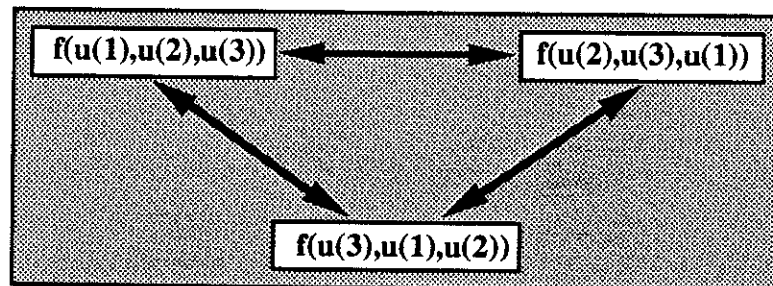


Figure 4 : Permutation cyclique d'indices d'ordre 3

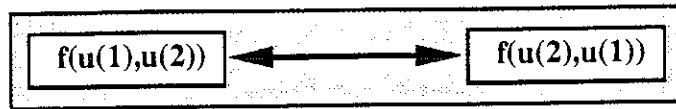


Figure 5 : Permutation cyclique d'indices d'ordre 2

De même, des paramètres algébriques (par opposition à physiques) peuvent être introduits dans les équations. Ainsi, les deux configurations :

$$\begin{array}{ll} i_1 = i_2 & \text{et} & i_1 = -i_2 \\ V_1 = V_2 & & V_1 = -V_2 \end{array}$$

peuvent être réunies en une seule

$$\begin{array}{l} i_1 = a \cdot i_2 \\ V_1 = a \cdot V_2 \end{array}$$

où : $a = \pm 1$. "a" est affecté au cours de la simulation dynamique, de la même façon que "indice" dans le cas du redresseur triphasé. Afin d'éviter des risques de division par 0, incontrôlables en calcul formel, il faut éviter d'utiliser des paramètres d'équation pouvant être nuls.

2) NOMENCLATURE

Pour structurer l'écriture des blocs, des fichiers "squelettes" de blocs ont été réalisés et sont à la disposition des utilisateurs, pour chaque type de bloc.

De même, une nomenclature a été définie pour caractériser les grandeurs utilisées dans les blocs. Ceci se fait par la deuxième lettre des noms, par exemple, un "P" signifie que la grandeur est un paramètre et un "E" que la grandeur est une entrée du bloc.

La première lettre des grandeurs des blocs caractérise leur nature, c'est-à-dire :

- I pour integer,
- D pour double précision,
- R pour real,
- K pour commentaire, caractère.

L'affectation de cette lettre est aisée pour la commande numérique où c'est à l'utilisateur d'écrire l'algorithme du séquençement et de l'enchaînement des blocs en Fortran. De plus, des

indications lui sont fournies par les blocs qu'il utilise et qui sont stockés en bibliothèque. Pour la puissance et la commande analogique, les grandeurs qui interviennent dans les équations auront D pour première lettre. Celles utilisées par les sous-programmes de gestion de configuration respecteront la nomenclature.

Attention, cette dernière doit être scrupuleusement respectée, car elle est nécessaire au bon fonctionnement du générateur. Notons que, pour le type des blocs :

- P signifie puissance,
- TS signifie commande,
- TSa signifie commande analogique,
- TSn signifie commande numérique.

L'origine de l'abréviation TS provient de Traitement du Signal.

3) ENTREE DES DONNEES

Les pages suivantes présentent le fichier d'entrée des données, dans le cas de l'application traitée au chapitre IV.A (figure 6).

Tous les commentaires sont situés entre "/*" et "*/". Ce fichier est composé essentiellement de :

- la déclaration des grandeurs du problème étudié,
- la liste des grandeurs à tracer,
- la déclaration des équations de puissance,
- la définition des grandeurs de puissance (variable d'état, de sortie, d'excitation, ...).

Dans la suite, nous présentons l'algorithme de commande (figure 7), utilisant les blocs (sous-programmes) disponible en bibliothèques : CANB , filtre , horloge , régulateur.

Cet algorithme se termine par l'appel des blocs (sous-programmes) de gestion des configurations des convertisseurs statiques : hacheur 4 quadrants commandé en MLI et pont redresseur à diodes . Ces blocs sont aussi disponibles en bibliothèque.

```

racine_nom_fichier:"CSTEPH"$
numero_module_commande:1$
val_knom:"'MCOH : machine cc alimente par un hacheur a MLI'"$
il_y_a_une_machine:false$
liste_grandeur_commande:
[[DPUR,"tension de declenchement du transi alim (V)",300.0E0],
[DPTMC,"periode de commutation du hacheur (s) ",63.75E-6],
[DPBTR,"temps garantissant le blocage des inter (s)",
1.0E-6],
[ICT,"commande des transistors du hacheur",[0,0,0,0]],
[IVHAC,"indicateur de configuration du hacheur",0],
[DTREF,"reference de courant (A)",10.0E0],
[RPCONA,"constante A du regulateur PI",0.515E0],
[RPCONB,"constante B du regulateur PI",0.395E0],
[RPTCA1,"temps d execution du calcul de sortie (s)",100.0E-6],
[RPTCA2,"temps d execution du calcul reccurent (s)",100.0E-6],
[RPREF,"reference de courant (A)",10.0E0],
[DPBETA,"sortie du regulateur",0.0E0],
[IPBUTE,"butee de sortie du regulateur",0],
[IPCNL,"correction de non linearite du hacheur",0],
[ITREF,"reference numerique du regulateur",0],
[ITGMIN,"butee de sortie min du canb",0],
[ITGMAX,"butee de sortie max du canb",0],
[IVMEM,"memoires internes du regulateur",[0,0,0,0]],
[ISGS,"beta de commande numerise",0],
[ISINUM,"valeur numerique du courant a reguler",0],
[ISEPS,"erreur numerique de la regulation",0],
[IOUREC,"terme recurrent du regulateur",0],
[DVMENT,"memoire interne du regulateur",0.0E0],
[IFC1,"impulsion fin de calcul du beta de commande",0],
[DOALFA,"alpha du hacheur =(beta+1)/2",0.0E0],
[DPPERI,"periode d echantillonnage de la regulation (s)",
318.75E-6],
[ISSYNC,"impulsion d horloge pour le canb",0],
[RPCAL,"calibre du canb (A)",25.0E0],
[RANVAL,"nombre aleatoire",0.0E0],
[RPBK,"bruit sur le courant",0.0E0],
[RANRES,"bruit sur la tension",0.0E0],
[IPRESO,"resolution du canb du regulateur (b)",12],
[ITGSMA,"butee de sortie max du regulateur",0],
[ITGSMI,"butee de sortie min du regulateur",0],
[IFCALC,"indicateur de fin de conversion du canb",0],
[DTQUAN,"quantum du canb",0.0E0],
[DVGRE,"grandeur tronquee par le canb",0.0E0],
[DVMEM1,"memoire interne de 1 horloge",0.0E0]]$

```

```

liste_grandeur:
[[DPVRC,"tension crete a crete du reseau (modifiee) (V)",
80.0E0],
[DPFREQ,"frequence reseau (Hz)",50.0E0],
[omegar,"pulsation du reseau",D2PIC*DPFREQ],
[dsures,"tension reseau",
[DPVRC*sin(omegar*dtc),
DPVRC*sin(omegar*dtc+d2ps3c),
DPVRC*sin(omegar*dtc-d2ps3c)]],
[ILREDR,"indicateur de configuration du redresseur",0],
[isindred,"indice des tensions reseau pour le redres",
[1,2,3]],
[DSUR,"tension redressee",0.0E0],
[DSIR1,"courant en sortie de redresseur",0.0E0],
[DSDIR1,"derivee du courant DSIR1",0.0E0],
[DSUS,"tension aux bornes de C1",132.0E0],
[DSDUS,"derivee de la tension Uc",0.0E0],
[DSIS,"courant en entree du hacheur",0.0E0],
[DPRF,"resistance de fuite de 1 alim (Ohm)",1.0E05],

```

```

[DPRP,"resistance de protection de l'alim (Ohm)",5.0E0],
[DPL1,"inductance de l'alimentation L (H)",10.0E-6],
[DPRL1,"resistance de l'alimentation (Ohm)",0.15E-0],
[DPC1,"capacite C1 (F)",47.0E-06],
[DPC2,"capacite C2 (F)",1.5E-3],
[DUREQ,"resistance equivalente du filtre d'alim",0.0E0],
[DPED,"fem due a la diode (V)",0.6E0],
[DPET,"fem due au transistor (V)",0.6E0],
[DPRD,"resistance de la diode passante (Ohm)",0.1E0],
[DPRT,"resistance du transistor passant (Ohm)",0.1E0],
[DSUL,"tension d'alimentation de la machine",0.0E0],
[DSIL,"courant machine",0.0E0],
[DSDIL,"derivee du courant machine",0.0E0],
[RKAMLI,"parametre de calcul du hacheur",1.0E0],
[ILHMLI,"indicateur de configuration du hacheur",0],
[DPL,"inductance totale de la machine cc (H)",1.15E-3],
[DPR,"resistance totale de la machine cc (Ohm)",1.18],
[DPK,"coefficient de couple (Nm)",0.245],
[DPEB,"chute de tension aux balais de la machine (V)",0.5],
[DSEB,"chute de tension signee aux balais",0.0E0]]$
liste_a_tracer:[DSUR,DSUS,DSIL,DPBETA,ICT[1],
ICT[2],ICT[3],ICT[4]]$
liste_a_tracer_p:["DSUR","DSUS","DSIL","DPBETA","ICT[1]",
"ICT[2]","ICT[3]","ICT[4]]$
eq11:[dgamc=dpk*dsil,
      dsdl=(-dpr*dsil-dpk*dvitr+dsul-dseb)/dpl,
      dpl1*dsdl=(dsur-dsus-dsdl*dprl),
      dsds=(dsdl-dsis-dsus*dureq)/(dpc1+dpc2)]$
equations_de_sortie:[]$
excitation:[DSURES[1],DSURES[2],DSURES[3],dseb,dvitr]$
vardle:[dsil,dsdl,dsds]$
derile:[dsdl,dsdl,dsds]$
varile:[dgamc,dsul,dsis,dsur]$
nombre_de_convertisseurs:2$
array(convertisseur,nombre_de_convertisseurs)$
convertisseur[1]:
  [ilredr,[1,[dsdl=0],[],[[[]],
    [2,[dsur=dsures[1]-dsures[3]],[isindred,3,dsures],[[]]]$
conditions_de_cond_dis[1]:[1,[dsdl=0]]$
convertisseur[2]:
  [ilhml,[[1,[dsul=rkamli*(dpet+dped)-(dprt+dprd)*dsil,dsis=0],[[[]],
    [2,[dsul=rkamli*(dsus-2*dpet-2*dprt*dsil),dsis=rkamli*dsil],
    [3,[dsul=rkamli*(dsus+2*dped+2*dprd*dsil),dsis=rkamli*dsil],
    [4,[dsil=0,dsis=0],[[[]]]$
conditions_de_cond_dis[2]:[4,[dsil=0]]$
batchload("conddis.ma7")$

```

Figure 6 : Fichier d'entrée des données pour l'exemple du chapitre IV.A

```

C
C Sequencement de la boucle de commande :
C
C gestion de la configuration du redresseur triphase :
C
    CALL XPTDIOD (DSURES,DSUS,DSIR1,
    +             ILREDR,ISIREL)
C
C Conversion analogique / numerique du courant de reference regule :
C
    CALL XCANB (1,DTREF,ITGMAX,ITGMIN,DTQUAN,
    &           IIBID,ITREF,DVGRE)
C
C Filtrage :
C
    CALL XFILTRE (DPUR,DSUS,DPRF,DPRF,DUREQ)
C
C On reactualise les variables de la boucle de regulation tout
C les 5*To
C
C horloge :
C
    CALL XHORLO (DPPERI,DTC,DVMEML,ISSYNC)
C
C Conversion analogique / numerique du courant regule :
C
    CALL XCANB ( ISSYNC,DSIL,ITGMAX,ITGMIN,DTQUAN,
    &           IFCALC,ISINUM,DVGRE)
C
C regulation PI
C
    CALL XREGPI (IFCALC,ITREF,ISINUM,IPCNL,RPCONA,RPCONB,RPTCA1,
    &           RPTCA2,ITGSMA,ITGSMI,DTC, IOUREC,IVMEM,DVMEML,
    &           IFC1,ISGS,ISEPS)
C
C Le ISGS sortant du regulateur PI etant le beta numerise on suit sur
C un convertisseur numerique analogique
C
C Calcul du BETA
C
    CALL XCNA ( IFC1,IPRESO,ISGS,
    +         IIFCAL,DSBETA )
C
C gestion des configurations du hacheur
C
    CALL XHACHEUR ( DSBETA,DSIL,DTC,DPTMC,
    &           DPBTR,ICT,
    &           DOALFA,IVHAC,ILHMLI,DKAMLI)
C
C calcul de la chute de tension aux balais de la machine
C
    CALL XMACCC (DSIL,DPEB,DSEB)
C

```

Figure 7 : Algorithme de la commande de l'exemple du chapitre IV.A

ANNEXE IV

CODE PRODUIT PAR NOTRE GENERATEUR POUR MC2

1) PRESENTATION DU CODE PRODUIT PAR LE GENERATEUR

Dans cette annexe, nous reprenons l'exemple traité au chapitre IV.A et dans l'annexe III.

Une part importante de la programmation faite automatiquement par le générateur concerne l'initialisation des données, la visualisation et la modification interactives des paramètres, la sauvegarde des données et des résultats, et la préparation des tracés graphiques de résultats.

Pour l'exemple du chapitre IV.A, nous présentons uniquement deux des huit configurations réorganisées et minimisées pour une intégration différentielle numérique.

Les trois formes de réorganisations présentées au § II.C.3.d ont été réalisées. A noter, pour la forme (3), que DRESGA est un sous-programme de résolution numérique de système linéaire (GAUSS).

On peut constater que la taille du système d'état est variable selon les configurations de l'association.

```

C FORME (1) : GAUSS formel
C -----
C
C obtention de dX1/dt
C *****
C
      IF( ILREDR .EQ. 1 ) THEN
        IF( ILHMLI .EQ. 4 ) THEN
          DSDUS = -1.0*DSUS*DUREQ/(DPC2+DPC1)
          DERY(1) = DSDUS
          DERY(2) = 0.0
          DERY(3) = 0.0
          DY(2) = 0.0
          DY(3) = 0.0
        ENDIF
      ENDIF
C
      IF( ILREDR .EQ. 2 ) THEN
        IF( ILHMLI .EQ. 1 ) THEN
          DSDIL = -1.0*(DPK*DVITRC+(DPRT+DPRD+DPR)*DSIL+DSEB-1.0*DKAMLI
1          *DPET-1.0*DKAMLI*DPED)/DPL
          DSDIR1 = -1.0*(DSUS+DPRL1*DSIR1+DSURES(ISIRED(3))-1.0*DSURES(
1          ISIRED(1)))/DPL1
          DSDUS = -1.0*(DSUS*DUREQ-1.0*DSIR1)/(DPC2+DPC1)
          DERY(1) = DSDIL
          DERY(2) = DSDIR1
          DERY(3) = DSDUS
        ENDIF
      ENDIF

```

Annexe IV : Code produit par notre générateur pour MC2

```

C
C avec les calculs séquentiels suivants
C
C obtention de Y1
C *****
C
      IF( ILREDR .EQ. 1 ) THEN
        IF( ILHMLI .EQ. 4 ) THEN
          DGAMC = 0.0
          DSUL = DPK*DVITRC+DSEB
          DSUR = DSUS
          DSIR1 = 0.0
          DSIL = 0.0
          DSDIR1 = 0.0
          DSDIL = 0.0
          DSIS = 0.0
        ENDIF
      ENDIF
C
      IF( ILREDR .EQ. 2 ) THEN
        IF( ILHMLI .EQ. 1 ) THEN
          DGAMC = DPK*DSIL
          DSUL = (-1.0*DPRT-1.0*DPRD)*DSIL+DKAMLI*DPET+DKAMLI*DPED
          DSUR = DSURES(ISIRED(1))-1.0*DSURES(ISIRED(3))
          DSIS = 0.0
        ENDIF
      ENDIF
C
C -----
C
C FORME (2) : triangularisation
C -----
C
C obtention de dX1/dt et Y1
C *****
C
      IF( ILREDR .EQ. 1 ) THEN
        IF( ILHMLI .EQ. 4 ) THEN
          DSDUS = -1.0*DSUS*DUREQ/(DPC2+DPC1)
          DGAMC = 0.0
          DSUL = DPK*DVITRC+DSEB
          DSUR = DSUS
          DERY(1) = DSDUS
          DERY(2) = 0.0
          DERY(3) = 0.0
          DY(2) = 0.0
          DY(3) = 0.0
        ENDIF
      ENDIF
C
      IF( ILREDR .EQ. 2 ) THEN
        IF( ILHMLI .EQ. 1 ) THEN
          DSDIL = (-1.0*DPK*DVITRC+(-1.0*DPRT-1.0*DPRD-1.0*DPR)*DSIL-1.
1          0*DSEB+DKAMLI*DPET+DKAMLI*DPED)/DPL
          DSDIR1 = (-1.0*DSUS-1.0*DPRL1*DSIR1-1.0*DSURES(ISIRED(3))+DSU
1          RES(ISIRED(1)))/DPL1
          DSDUS = (DSIR1-1.0*DSUS*DUREQ)/(DPC2+DPC1)
          DGAMC = DPK*DSIL
          DSUL = DPK*DVITRC+DPR*DSIL+DSEB+DPL*DSDIL
          DERY(1) = DSDIL
          DERY(2) = DSDIR1
          DERY(3) = DSDUS
        ENDIF
      ENDIF

```

```

C
C avec les calculs séquentiels suivants
C
C obtention de Y2
C *****
C
      IF( ILREDR .EQ. 1 ) THEN
        IF( ILHMLI .EQ. 4 ) THEN
          DSIR1 = 0.0
          DSIL = 0.0
          DSDIR1 = 0.0
          DSDIL = 0.0
          DSIS = 0.0
        ENDIF
      ENDIF
C
      IF( ILREDR .EQ. 2 ) THEN
        IF( ILHMLI .EQ. 1 ) THEN
          DSUR = DSURES(ISIRED(1))-1.0*DSURES(ISIRED(3))
          DSIS = 0.0
        ENDIF
      ENDIF
C
C -----
C
C FORME (3) : GAUSS numerique
C -----
C
C obtention de dx1/dt et Y1
C *****
C
      IF( ILREDR .EQ. 1 ) THEN
        IF( ILHMLI .EQ. 4 ) THEN
          GAUSSN= 4
          DO 17 GAUSI1=1, 4
            DO 16 GAUSI2=1, 4
              GAUSSA(GAUSI1,GAUSI2)=0.0D0
16          CONTINUE
              GAUSSB(GAUSI1)=0.0D0
17          CONTINUE
              GAUSSA(1,2) = 1.0
              GAUSSA(2,3) = -1.0/DPL
              GAUSSA(3,4) = -1.0
              GAUSSA(4,1) = 1.0
              GAUSSB(2) = -1.0*(DPK*DVITRC+DSEB)/DPL
              GAUSSB(3) = -1.0*DSUS
              GAUSSB(4) = -1.0*DSUS*DUREQ/(DPC2+DPC1)
              CALL DRESGA(GAUSSA,GAUSSN,GAUSSM,
&              GAUSSB,GAUSSE,GAUSSX,GAUSSI)
              IF (GAUSSI.EQ.1) THEN
                CALL ECRMSG('systeme insolvable')
                CALL STOPD1
              ENDIF
              DSDUS = GAUSSX(1)
              DGAMC = GAUSSX(2)
              DSUL = GAUSSX(3)
              DSUR = GAUSSX(4)
              DERY(1) = GAUSSX(1)
              DERY(2) = 0.0
              DERY(3) = 0.0
              DY(2) = 0.0
              DY(3) = 0.0
            ENDIF
          ENDIF
C

```

```

IF( ILREDR .EQ. 2 ) THEN
  IF( ILHMLI .EQ. 1 ) THEN
    GAUSSN= 5
    DO 19 GAUSI1=1, 5
      DO 18 GAUSI2=1, 5
        GAUSSA(GAUSI1,GAUSI2)=0.0D0
18      CONTINUE
        GAUSSB(GAUSI1)=0.0D0
19      CONTINUE
        GAUSSA(1,5) = 1.0
        GAUSSA(2,4) = 1.0
        GAUSSA(3,1) = 1.0
        GAUSSA(3,5) = -1.0/DPL
        GAUSSA(4,2) = DPL1
        GAUSSA(5,3) = 1.0
        GAUSSB(1) = (-1.0*DPRT-1.0*DPRD)*DSIL+DKAMLI*DPET+DKAMLI*DPED
        GAUSSB(2) = DPK*DSIL
        GAUSSB(3) = -1.0*(DPK*DVITRC+DPR*DSIL+DSEB)/DPL
        GAUSSB(4) = -1.0*DSUS-1.0*DPRL1*DSIR1-1.0*DSURES(ISIRED(3))+D
1        SURES(ISIRED(1))
        GAUSSB(5) = -1.0*(DSUS*DUREQ-1.0*DSIR1)/(DPC2+DPC1)
        CALL DRESGA(GAUSSA,GAUSSN,GAUSSM,
&        GAUSSB,GAUSSE,GAUSSX,GAUSSI)
        IF (GAUSSI.EQ.1) THEN
          CALL ECRMSG('systeme insolvable')
          CALL STOPD1
        ENDIF
        DSDIL = GAUSSX(1)
        DSDIR1 = GAUSSX(2)
        DSDUS = GAUSSX(3)
        DGAMC = GAUSSX(4)
        DSUL = GAUSSX(5)
        DERY(1) = GAUSSX(1)
        DERY(2) = GAUSSX(2)
        DERY(3) = GAUSSX(3)
      ENDIF
    ENDIF
C
C avec les calculs séquentiels suivants
C
C obtention de Y2
C *****
C
IF( ILREDR .EQ. 1 ) THEN
  IF( ILHMLI .EQ. 4 ) THEN
    DSIR1 = 0.0
    DSIL = 0.0
    DSDIR1 = 0.0
    DSDIL = 0.0
    DSIS = 0.0
  ENDIF
ENDIF
C
IF( ILREDR .EQ. 2 ) THEN
  IF( ILHMLI .EQ. 1 ) THEN
    DSUR = DSURES(ISIRED(1))-1.0*DSURES(ISIRED(3))
    DSIS = 0.0
  ENDIF
ENDIF
C

```

Figure 1 : Configurations 1/4 et 2/1

2) SIMPLIFICATIONS DES EQUATIONS

Afin que le générateur puisse simplifier au mieux les équations qu'il réorganise, des procédures et des fonctions particulières (écrites en langage Macsyma) ont dû être réalisées. Elles concernent essentiellement :

- des règles mathématiques de simplification d'expressions trigonométriques, issues des modèles de machines électriques tournantes,
- la création de variables intermédiaires de calcul, qui remplacent par une grandeur toute expression que l'on retrouve de nombreuses fois dans les équations. Par exemple, la variable S2T2P3 est créée et substituée à $\sin(2\omega t + 2\pi/3)$ dans l'ensemble des équations réorganisées. Pour cela, il a fallu veiller à ce que le générateur puisse reconnaître les identités suivantes :

$$\begin{aligned}
 \sin(2\omega t + 2\pi/3) &= \sin(2(\omega t + \pi/3)) \\
 &= \sin((6\omega t + 2\pi)/3) \\
 &= \sin(2(3\omega t + \pi)/3) \\
 &= \dots\dots\dots
 \end{aligned}$$

- des constantes propres à MC2, par exemple :

π	$\rightarrow$	DPIC	dans MC2
$\pi/3$	$\rightarrow$	DPIS3C	dans MC2
$\sqrt{3}$	$\rightarrow$	DRAC3C	dans MC2
$\sqrt{3}/2$	$\rightarrow$	DR3S2C	dans MC2

qui sont les grandeurs que l'on retrouve souvent dans les équations utilisées en génie électrique.

Afin de pouvoir générer des expressions en Fortran de taille importante, nous avons élaboré des procédures adéquates. De même, comme pour les parties puissances et commande analogique des associations, nous avons choisi de faire les calculs en double précision, nous avons élaboré des fonctions qui permettent d'assurer ceci dans les programmes générés. Par exemple, la fonction sinus s'écrit DSIN et non SIN qui est utilisé pour les calculs en real).

Notons que c'est grâce à l'expérience acquise sur Macsyma et sur les équations manipulées en électrotechnique, que ces simplifications ont pu être réalisées et optimisées.

```
C fichier CALCTRIG.FOR
C
C
C
C
C      fichier de calcul des variables
C          trigonometriques
C
C
SUBROUTINE CALCTRIG
C
C $INCLUDE:'COMTRIG1.INC'
C $INCLUDE:'NUMCONS1.INC'
C $INCLUDE:'CFELIPE.INC'
C
DOUBLE PRECISION TETAE
C affectation de l'angle electrique
TETAE=DSTETA
C calcul des variables internes declarees ci-dessus
C
SINT=DSIN( TETAE )
SIN2T=DSIN( 2.0D0 * TETAE )
SIN3T=DSIN( 3.0D0 * TETAE )
SIN4T=DSIN( 4.0D0 * TETAE )
ST2P3=DSIN( TETAE + D2PS3C )
S2T2P3=DSIN( 2.0D0 * TETAE + D2PS3C )
S3T2P3=DSIN( 3.0D0 * TETAE + D2PS3C )
S4T2P3=DSIN( 4.0D0 * TETAE + D2PS3C )
ST4P3=DSIN( TETAE - D2PS3C )
S2T4P3=DSIN( 2.0D0 * TETAE - D2PS3C )
S3T4P3=DSIN( 3.0D0 * TETAE - D2PS3C )
S4T4P3=DSIN( 4.0D0 * TETAE - D2PS3C )
COST=DCOS( TETAE )
COS2T=DCOS( 2.0D0 * TETAE )
COS3T=DCOS( 3.0D0 * TETAE )
COS4T=DCOS( 4.0D0 * TETAE )
CT2P3=DCOS( TETAE + D2PS3C )
C2T2P3=DCOS( 2.0D0 * TETAE + D2PS3C )
C3T2P3=DCOS( 3.0D0 * TETAE + D2PS3C )
C4T2P3=DCOS( 4.0D0 * TETAE + D2PS3C )
CT4P3=DCOS( TETAE - D2PS3C )
C2T4P3=DCOS( 2.0D0 * TETAE - D2PS3C )
C3T4P3=DCOS( 3.0D0 * TETAE - D2PS3C )
C4T4P3=DCOS( 4.0D0 * TETAE - D2PS3C )
C
RETURN
END
C
C *****
C *
C *                               Fin du fichier
C *
C *****
```

Figure 2 : Variables générées pour une association avec une machine synchrone

ANNEXE V

MODELE DE MACHINE SYNCHROME AVEC AMORTISSEURS

1) DESCRIPTION

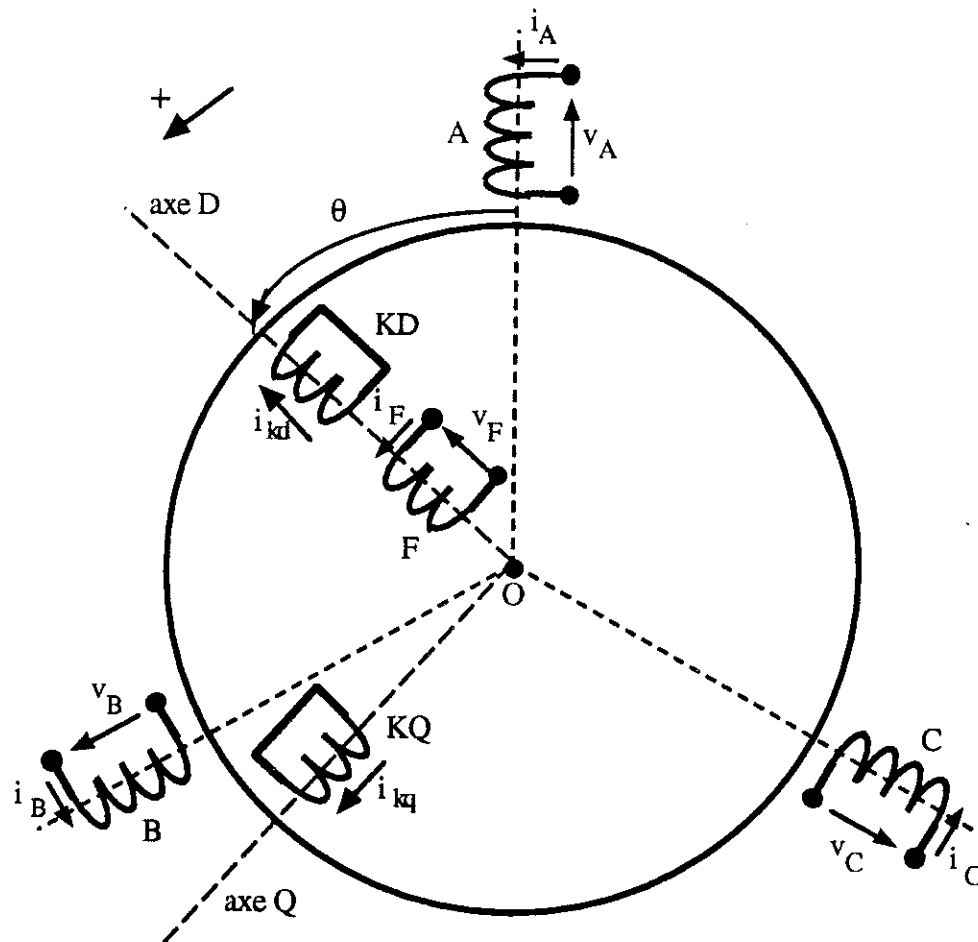


Figure 1 : Machine synchrone avec cage d'amortisseurs

La machine synchrone étudiée au Chapitre IV.B est caractérisée par :

- un inducteur tournant (rotor), à pôles saillants, de position θ , qui tourne à la vitesse Ω dans le sens trigonométrique,
- un entrefer variable, mais symétrique par rapport à deux axes perpendiculaires,
- un induit (stator) composé d'un enroulement triphasé A,B,C,
- des amortisseurs réalisés par un ensemble de barres conductrices logées dans les

encoches longitudinales du stator, au voisinage de la périphérie des pôles. Elles sont réunies aux extrémités par deux couronnes. Les amortisseurs s'opposent à toute variation rapide du flux à travers le rotor.

2) HYPOTHESES SIMPLIFICATRICES

- On suppose que le circuit magnétique n'est pas saturé, ce qui permet d'exprimer les flux comme des fonctions linéaires des courants.
- La force électromotrice est sinusoïdale.
- On admet que l'ensemble des amortisseurs peut être représenté par deux enroulements fermés en court-circuit sur eux-mêmes.

On néglige :

- l'effet de peau qui augmente les résistances et diminue les inductances,
- les couplages capacitifs entre enroulements,
- l'effet de la température sur la valeur des résistances.

3) EQUATIONS

$$R = R_A = R_B = R_C$$

$$M_{FD} = M_F, M_{FQ} = 0 \text{ et } M_{DQ} = 0$$

les équations de la machine s'écrivent :

$$[V_S] = [R_S] \cdot [i_S] + \frac{d}{dt} \{ [L_{SS}] \cdot [i_S] + [M_{SR}] \cdot [i_R] \}$$

$$[V_R] = [R_R] \cdot [i_R] + \frac{d}{dt} \{ [M_{RS}] \cdot [i_S] + [L_{RR}] [i_R] \}$$

avec :

$$[V_S] = \begin{bmatrix} v_A \\ v_B \\ v_C \end{bmatrix}, [V_R] = \begin{bmatrix} v_F \\ 0 \\ 0 \end{bmatrix}, [i_S] = \begin{bmatrix} i_A \\ i_B \\ i_C \end{bmatrix}, [i_R] = \begin{bmatrix} i_F \\ i_{KD} \\ i_{KQ} \end{bmatrix}$$

$$[R_S] = R \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, [R_R] = \begin{bmatrix} R_F & 0 & 0 \\ 0 & R_{KD} & 0 \\ 0 & 0 & R_{KQ} \end{bmatrix}, [M_{RS}] = [M_{SR}]^t$$

$$[L_{SS}] = \begin{bmatrix} L_A & M_{AB} & M_{AC} \\ M_{AB} & L_B & M_{BC} \\ M_{AC} & M_{BC} & L_C \end{bmatrix}, [L_{RR}] = \begin{bmatrix} L_F & M_{FD} & 0 \\ M_{FD} & L_{KD} & 0 \\ 0 & 0 & L_{KQ} \end{bmatrix}$$

$$[M_{SR}] = \begin{bmatrix} M_{AF} & M_{AD} & M_{AQ} \\ M_{BF} & M_{BD} & M_{BQ} \\ M_{CF} & M_{CD} & M_{CQ} \end{bmatrix}$$

L'expression du couple électromagnétique s'écrit :

$$\Gamma_e = p \cdot \left\{ \frac{1}{2} [i_s]_t \cdot \left\{ \frac{d}{d\theta} [L_{SS}] \right\} \cdot [i_s] + [i_s]_t \cdot \left\{ \frac{d}{d\theta} [M_{SR}] \right\} \cdot [i_R] \right\}$$

Valeur des inductances

On a :

L_{SV} : amplitude des inductances statoriques fonctions de θ

$M_0 = -L_0/2$ constante

- inductances stator :

$$L_A = L_0 + L_{SV} \cdot \cos 2\theta$$

$$L_B = L_0 + L_{SV} \cdot \cos(2\theta + 2\pi/3)$$

$$L_C = L_0 + L_{SV} \cdot \cos(2\theta - 2\pi/3)$$
- mutuelles stator :

$$M_{AB} = M_0 + L_{SV} \cdot \cos(2\theta - 2\pi/3)$$

$$M_{BC} = M_0 / L_{SV} \cdot \cos 2\theta$$

$$M_{AC} = M_0 + L_{SV} \cdot \cos(2\theta + 2\pi/3)$$
- mutuelles stator-inducteur :

$$M_{AF} = M_{FS} \cdot \cos \theta$$

$$M_{BF} = M_{FS} \cdot \cos(\theta - 2\pi/3)$$

$$M_{CF} = M_{FS} \cdot \cos(\theta + 2\pi/3)$$
- mutuelles stator-cage :

$$M_{AD} = M_{DS} \cdot \cos \theta$$

$$M_{BD} = M_{DS} \cos(\theta - 2\pi/3)$$

$$M_{CD} = M_{DS} \cos(\theta + 2\pi/3)$$

$$M_{AQ} = -M_{QS} \sin \theta$$

avec l'axe direct

$$M_{BQ} = -M_{QS} \sin(\theta - 2\pi/3) \quad \text{avec l'axe quadrature}$$

$$M_{CQ} = -M_{QS} \sin(\theta + 2\pi/3)$$

3) VALEURS NUMERIQUES DES GRANDEURS

Les paramètres de la machine pour la simulation présentée au chapitre IV.B sont :

Stator :

- Résistance de chaque phase : $R_A = R_S = 0,113 \, \Omega$
- Résistance de l'enroulement d'excitation : $R_F = 41,0 \, \Omega$
- Inductance homopolaire : $L_0 = 4,72 \, \text{mH}$
- Inductance $L_{SV} = 1,01 \, \text{mH}$
- Inductance de l'enroulement d'excitation : $L_F = 24,6 \, \text{H}$
- Mutuelle $M_0 = -2,36 \, \text{mH}$
- Mutuelle inducteur-stator : $M_{FS} = 339 \, \text{mH}$

Rotor :

- Résistance amortisseur axe D : $R_D = 0,295 \, \Omega$
- Résistance amortisseur axe Q : $R_Q = 0,092 \, \Omega$
- Inductance homopolaire : $L_0 = 4,72 \, \text{mH}$
- Inductance axe D : $L_D = 64,9 \, \text{mH}$
- Inductance axe Q : $L_Q = 11,41 \, \text{mH}$
- Mutuelle cage-rotor : $M_{FD} = 1,22 \, \text{H}$
- Mutuelle stator-cage axe D : $M_{DS} = 17,9 \, \text{mH}$
- Mutuelle stator-cage axe Q : $M_{QS} = 5,55 \, \text{mH}$

ANNEXE VI

SIMULATION DE LA COMMANDE NUMERIQUE

1) NUMERISATION DES GRANDEURS

Dans la commande numérique, la numérisation entraîne une discrétisation (format) et des limitations des grandeurs utilisées. Par exemple, l'équation suivante, écrite avec des grandeurs physiques :

$$\varphi = L \cdot I$$

où φ : flux de l'inductance,

L : inductance,

I : courant dans l'inductance.

En numérique, elle devient :

$$X_{\varphi} = X_C \cdot X_I \quad \text{où}$$

$$X_C = \frac{(X_{\varphi})_{\max}}{(\varphi)_{\max}} \cdot \frac{I_{\max}}{X_{I_{\max}}} \cdot X_L$$

le coefficient X_C est un nombre constant

les grandeurs physiques φ , L et I sont respectivement représentées par les nombres X_{φ} , X_L et X_I .

Des limitations sont réalisées pour que X_{φ} (respectivement X_I) ne sorte pas de ses limites [$X_{\varphi \min}$, $X_{\varphi \max}$] (respectivement [$X_{I \min}$, $X_{I \max}$]).

Lors de la programmation de calculs avec ces grandeurs, il faut être prudent pour éviter que des nombres soient tronqués. Par exemple, on veille à ce que le produit de deux nombres codés sur 8 bits soit affecté dans un nombre codé sur 16 bits.

Dans notre approche, nous supposons que cela est vérifié, ce qui nous permet de ne prendre en compte la numérisation et les limitations des grandeurs qu'en entrées et sorties des blocs.

2) SIMULATION DE L'EVOLUTION EN TEMPS REEL DES FONCTIONS

Deux niveaux doivent être pris en compte pour définir les conditions d'exécution d'un algorithme d'une fonction de commande [FAR91].

a) Niveau fonctionnel

Dans ce cas, une validation du bloc apparaît (figure 1). Elle lui est externe et est liée à un séquenceur qui, selon les modes de fonctionnement de la commande, valide ou non certains blocs (c'est-à-dire fonctions).

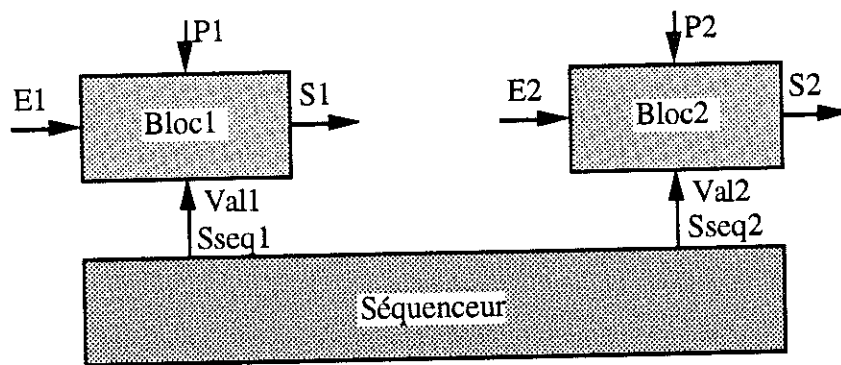


Figure 1 : Niveau fonctionnel des blocs TS_n

b) Niveau organique

Afin de prendre en compte l'organisation logicielle des fonctions de commande pendant un mode de fonctionnement, nous utilisons la notion d'activation.

Cette notion apparaît au niveau même des blocs, mais aussi au niveau de l'exécution des algorithmes des fonctions qui correspondent à des tâches.

Ainsi, afin de simuler le temps de calcul réel des fonctions (c'est à dire des blocs), on utilisera les trois notions suivantes :

- activation,
- durée d'exécution,
- fin de calcul.

Celles-ci nous permettent de représenter le fonctionnement de chaque bloc selon l'organigramme suivant (figure 2):

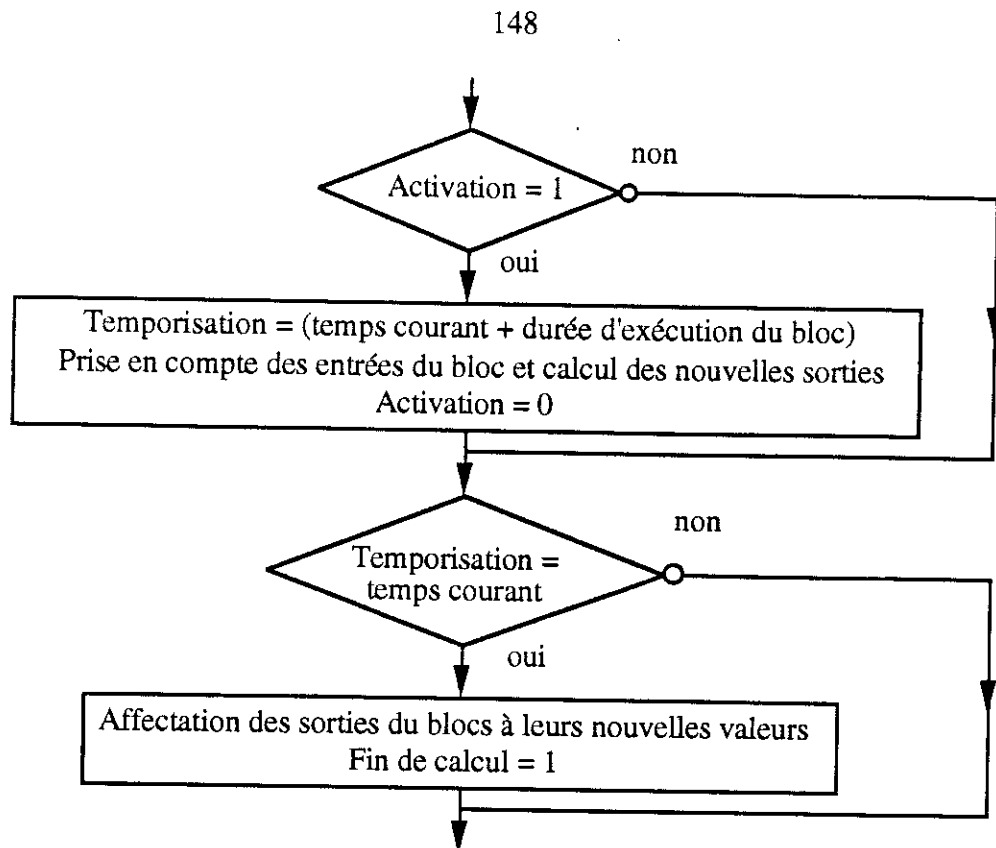


Figure 2 : Organigramme

3) ORGANISATION DES ACTIVATIONS DES TACHES

a) Les priorités

Pour optimiser les temps de calcul et l'organisation algorithmique des commandes numériques, on utilise la notion de tâches avec priorité, sur le banc d'essai [PET92]. Par exemple, on constate que pour les systèmes de contrôle de couple, on peut faire apparaître plusieurs groupes de fonctions dont les niveaux de contrainte temporelle sont différents. Ainsi, les périodes d'échantillonnage peuvent différer pour ces groupes fonctionnels. Dans le cas de la machine asynchrone autopilotée, trois tâches peuvent être définies, en allant de celle s'effectuant à la fréquence la plus élevée, à celle s'effectuant à la fréquence la plus faible :

- 1- l'autopilotage,
- 2- le contrôle du couple,
- 3- le contrôle de la vitesse.

Ce classement permet de leur affecter une priorité d'exécution, de façon à bien gérer les contraintes temporelles ainsi que les calculs. Dans le cas présent, la tâche 1 devra être exécutée en priorité dès qu'elle sera validée. Ce n'est qu'une fois achevée que l'on permettra, si elles

sont activées, l'exécution des tâches 2 et 3. Notons que le même type de priorité d'exécution de la tâche 1 par rapport aux tâches 2 et 3, existe entre les tâches 2 et la tâche 3 [FAR91].

Si on ne considère que les deux premières tâches (c'est-à-dire un système sans contrôle de vitesse), on peut obtenir le chronogramme suivant (figure 3).

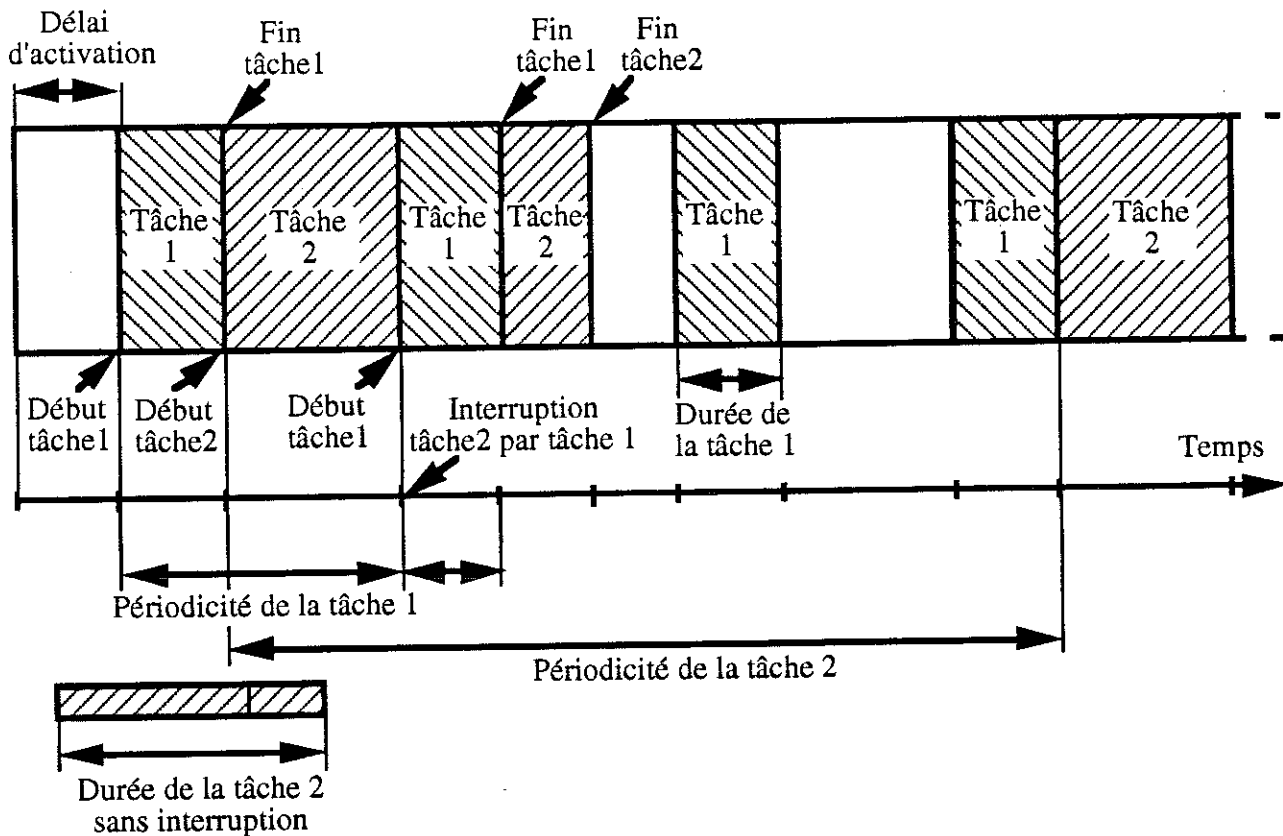


Figure 3 : Exemple de chronogramme pour un système à deux tâches

Deux notions importantes apparaissent :

- la tâche prioritaire,
- l'interruption d'une tâche.

Pour avoir une simulation du temps réel d'exécution de la commande (comme sur le banc d'essai), une gestion de priorités de tâches a été implantée dans MC2, par Ph. S. FARAH, au début de son DEA [FAR91]. Cette gestion se fait de façon transparente vis-à-vis de l'utilisateur.

b) Etat d'une tâche

La gestion des tâches se fait par un contrôleur de priorité (ou ordonnanceur) [PET92]. Ainsi,

les états d'une tâche peuvent être représentés par le GRACET suivant (figure 4).

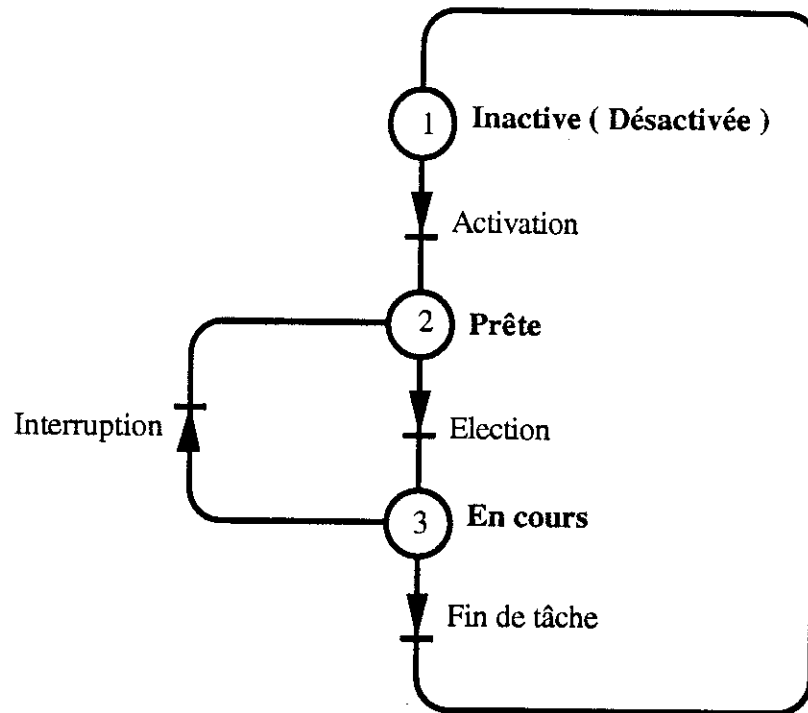


Figure 4 : Etat d'une tâche

Remarque : Rappelons qu'une tâche est constituée de plusieurs blocs et que la durée totale minimale d'exécution d'une tâche (c'est-à-dire sans interruption) est la somme de celles des différents blocs qui la constitue.

ANNEXE VII

DECOUPLAGE DES BOUCLES PAR DES BLOCS

Au chapitre II.D, afin d'optimiser les traitements symboliques faits par notre générateur et les traitements numériques de la simulation, nous avons montré qu'il est possible de découpler des boucles de puissance et de commande analogique par l'intermédiaire de blocs.

L'objectif de cette annexe est d'expliquer pourquoi un tel découplage est possible.

1) RAPPEL SUR LA RESOLUTION NUMERIQUE D'EQUATIONS DIFFERENTIELLES

Les explications qui suivent, sont basées sur une bonne compréhension du processus de résolution numérique de toute équation différentielle du type :

$$f(\dot{X}(t), X(t), Y(t), u(t), t) = 0$$

(avec les notations du chapitre II.C.3.b).

- * $u(t)$ est obtenu à partir du temps courant t ,
- * $\dot{X}(t)$ et $Y(t)$ sont obtenus à partir de $X(t)$ et $u(t)$,
- * $X(t)$ est obtenu à partir de $\dot{X}(t - \Delta t)$ (valeur de X au pas de calcul précédent) par un processus d'intégration numérique implicite (Runge Kutta).

2) REGLES DE DECOUPAGE

- * Tous les blocs qui fournissent uniquement des grandeurs constantes ou qui ne dépendent que du temps courant, permettent de découpler les boucles.
- * Toutes les boucles qui ne partagent pas de dérivées de variable d'état, peuvent être découplées. Ceci se fait par exemple, par les arbres des moteurs.

3) EXEMPLE

Pour illustrer ceci, prenons l'exemple de la figure 1. L'association présentée se compose :

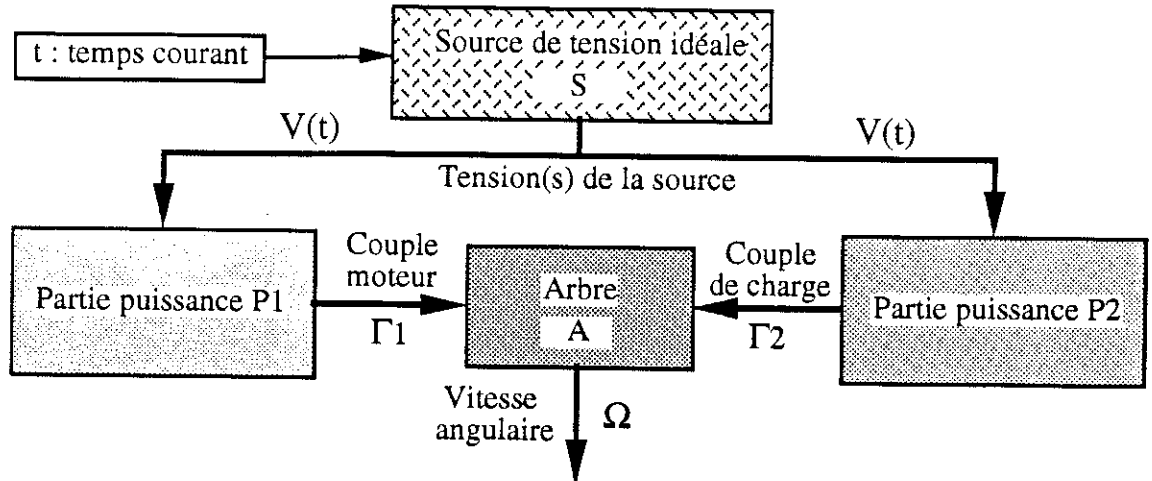


Figure 1 : Couplage direct par un arbre de deux parties puissance indépendantes vis à vis de la source de tension et de l'arbre

- d'une source de tension idéale (S) qui fournit la tension $V(t)$ en fonction du temps courant t ,
- de deux boucles de puissance P_1 et P_2 qui fournissent les couples $\Gamma_1(t)$ et $\Gamma_2(t)$; ces derniers dépendent de $\Omega(t)$ ou $\theta(t)$,
- d'un arbre rigide (A) d'équations :

$$J \cdot d\Omega/dt = \Gamma_1 - \Gamma_2 + C \cdot \Omega$$

$$\Omega = d\theta/dt$$

$$\gamma = d\Omega/dt$$
 où J : moment d'inertie et C : constante.

L'arbre fournit :

- la dérivée de la position angulaire $\theta(t)$: $d\theta(t)/dt$,
- la dérivée de la vitesse angulaire $\Omega(t)$: $d\Omega(t)/dt$,
- l'accélération angulaire $\gamma(t)$.

Dans ce cas, l'équation d'état globale, pour toute l'association, est du type :

$$f(\dot{X}(t), X(t), Y(t), u(t)) = 0$$

et est telle que :

- X contient : Ω, θ ,
- Y contient : $\Gamma_1, \Gamma_2, \gamma$,
- u contient : V .

A chaque pas de calcul Δt :

- * $V(t)$ est calculé à partir de t ,
- * $\Gamma_1(t)$, $\Gamma_2(t)$, $\gamma(t)$, $d\Omega(t)/dt$, $d\theta(t)/dt$ sont calculés à partir de $\Omega(t)$, $\theta(t)$ et $V(t)$,
- * $\theta(t)$ et $\Omega(t)$ sont obtenus par le processus d'intégration différentielle à partir de $d\Omega(t - \Delta t)/dt$ et de $d\theta(t - \Delta t)/dt$.

Comme les boucles P_1 , P_2 , A et S ne partagent aucune dérivée de variable d'état, une séparation de ces calculs est possible.

Il faut procéder ainsi à chaque pas de calcul :

- calcul de $V(t)$,
- puis calcul de $\Gamma_1(t)$, $\Gamma_2(t)$,
- puis calcul de $d\theta(t)/dt$, $d\Omega(t)/dt$, $\gamma(t)$.

On peut donc construire quatre petits sous-systèmes, qui sont résolus séparément et séquentiellement dans l'ordre (S, P_1 , P_2 , A).

ANNEXE VIII

MODES D'EXECUTIONS DES REGLES

1) PARAMETRISATIONS POSSIBLES

Dans Smeci, pour chaque tâche, il est possible de paramétrer la façon dont les règles s'exécutent.

a) L'instanciation des règles

- **toute** : application de toutes les règles instanciables sur toutes les combinaisons d'objets instanciables.
- **une seule règle-toute** : application d'une seule règle instanciable, sur toutes les combinaisons d'objets instanciables.
- **une seule règle-une** : application d'une seule règle instanciable sur une seule combinaison d'objets instanciables.

b) Le mode d'exécution

- **sans ordre** : l'exécution des règles se fait dans un ordre quelconque, le moteur d'inférence examine l'ensemble des règles.
- **dans l'ordre** : les règles sont appliquées dans l'ordre fournit par l'utilisateur, lors de la description de la tâche. Si la règle R_i n'est pas instanciable, le moteur d'inférence passe à la suivante : $R_i + 1$, et aucun retour en arrière n'est possible pour réappliquer R_i .
- **dans l'ordre stricte** : signifie que les règles sont appliquées comme précédemment, mais une seule fois.

c) Le mode de branchement

Le mode caractérise le processus de création des états :

- **séquentiel** : ce mode interdit l'application en parallèle d'une même règle, sur plusieurs jeux d'instanciation.
- **parallèle** : ce mode permet de gérer des raisonnements en parallèle, donc de conduire à plusieurs solutions de façon indépendante, en parallèle.

2) PARAMETRISATIONS UTILISEES DANS NOTRE SYSTEME D'ETUDE A LA CONCEPTION

Autour d'exemples très simples, ce paragraphe présente les combinaisons des paramètres précédents que nous avons utilisés pour gérer l'exécution des règles.

a) Notations

Notre exemple repose sur deux règles utilisées au sein d'une même tâche, et pour lesquelles on donne les paramètres de contrôle que nous avons utilisés. Les objets manipulés ont deux propriétés : une forme géométrique (carré ou cercle) et une couleur (noir ou blanc). Les états sont représentés par un rectangle horizontal où les objets sont dessinés.

b) Tâche purement séquentielle

Nous utilisons ce type de tâche, lorsque nous souhaitons obtenir un raisonnement purement séquentiel. Dans ce cas, nous affectons les paramètres de contrôle des règles, aux valeurs suivantes :

- instanciation : une seule règle toute,
- mode d'exécution : dans l'ordre stricte,
- mode de branchement : séquentiel.

Pour illustrer ceci, prenons l'exemple d'une tâche avec deux règles R_1 et R_2 fournies dans l'ordre (R_1 , R_2). La règle R_1 transforme les carrés en cercles (quelque soit leur couleur). La règle R_2 peint les objets blancs en noir. Avec les affectations précédentes pour les paramètres de contrôle des règles, on obtient l'arbre de raisonnement suivant (figure 1).

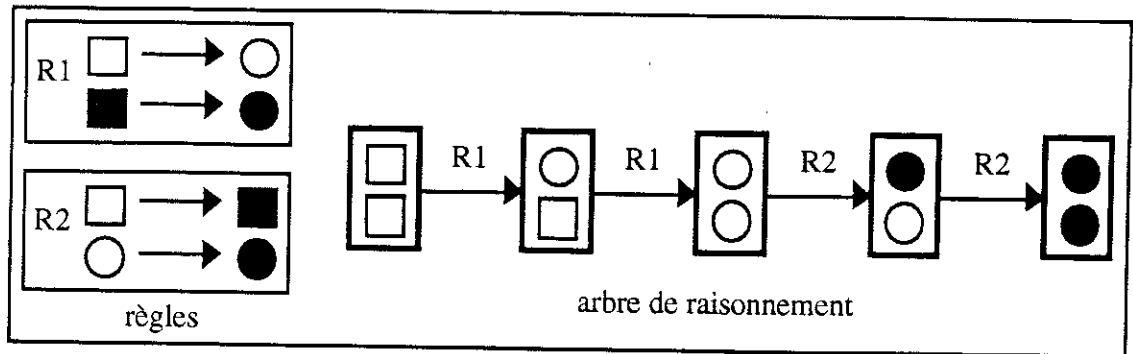


Figure 1 : Tâche purement séquentielle

c) Tâche purement parallèle (ou)

Nous utilisons ce type de tâche pour les règles contenant un "ou" dans leur partie action. Dans ce cas, nous affectons les paramètres de contrôle des règles, aux valeurs suivantes :

- instanciation : toute,
- mode d'exécution : dans l'ordre stricte,
- mode de branchement : parallèle.

et chaque partie action des règles se termine par l'instruction : \$(S-pop-task).

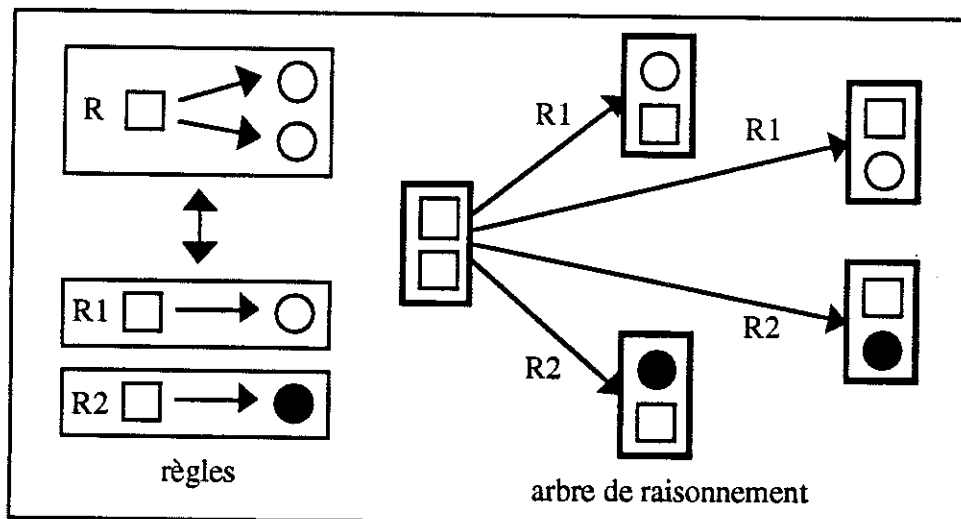


Figure 2 : Tâche purement parallèle (ou)

Illustrons ceci autour d'un exemple. Prenons la règle R qui donne deux solutions possibles : l'une transforme les carrés blancs en cercles blancs (R₁) et l'autre transforme les carrés blancs en cercles noirs (R₂). On construit une tâche contenant les deux règles R₁ et R₂ dans l'ordre (R₁, R₂). Chaque règle se termine par \$(S-pop-task) et la tâche possède les affectations

précédentes pour les paramètres de contrôle des règles. On obtient l'arbre de raisonnement suivant (figure 2).

d) Tâche purement parallèle (une)

Nous avons aussi utilisé les affectations suivantes pour une tâche ayant une règle devant donnée une solution pour chaque objet en parallèle. Dans ce cas, nous affectons les paramètres de contrôle des règles, aux valeurs suivantes :

- instantiation : une seule règle-toute,
- mode d'exécution : dans l'ordre stricte,
- mode de branchement : parallèle.

Illustrons ceci par un exemple. Soit une tâche qui contient une seule règle R1, avec les affectations précédentes pour le contrôle des règles. La règle R1 transforme un carré en rond. On obtient donc l'arbre de raisonnement suivant (figure 3).

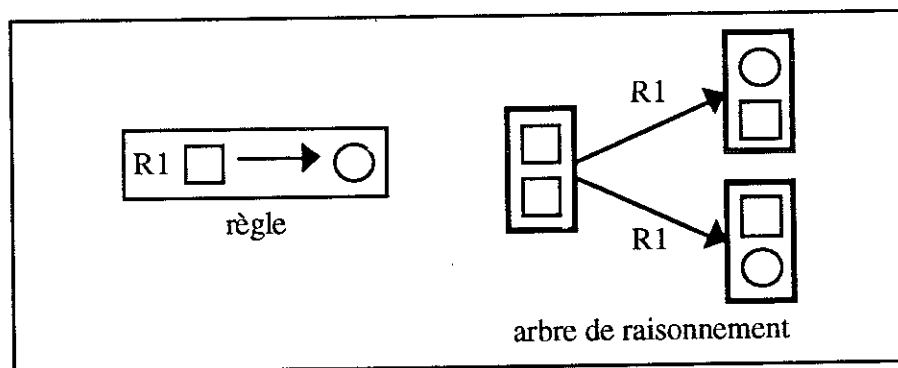


Figure 3 : Tâche purement parallèle (une)

Bibliographie

BIBLIOGRAPHIE

- [AHR91] S. AHRENS, J. KORTABERRIA, J. BIGEON, D. POUPART
"Expert system for designing converters for large synchronous machines".
4th European Conference on Power Electronics and Application, Firenze, EPE'91,
3-6 September 1991, Vol.2, pp 266-271.
- [ALG91] Recueil des premières conférences européennes sur "Algebraic computing and control". 13-15 Mars 1991, Paris.
- [BEZ92] G. BEZANOV, J. RICHARDSON
"Expert systems considerations for optimal PWM A.C. drives".
27th Power Engineering Conference 1992, University of Bath, Claverton Down, Bath,
23-25 September 1992, Vol 2, pp 396-399.
- [BOL89] A. BOLOPION
"Notice d'utilisation de MC2".
LEG, Juillet 1989.
- [BOL90] A. BOLOPION, D. JOUVE, R. PACAUT
"Control of permanent magnets synchronous machines : a simulation comparative survey".
Proceeding of IEEE-APEC'90, March 11-16, 1990, Los Angeles, California,
pp 374-383.
- [BOS86] B.K. BOSE
"Power electronics and AC drives".
Prentice Hall, 1986.
- [BOS89] B.K. BOSE
"Power electronics - an emergency technology".
IEEE Transaction on Industrial Electronics, vol 36, n°3, August 1989, pp 403-412.

- [BOS90] P.P.J. WAN DEN BOSCH, H.R. VISSER
 "Simulation of state-events in power electronic drives".
 4th Conference on Power Electronics and Variable Speed Drive, London, July 1990,
 pp 184- 189.
- [BOS92] B.K. BOSE
 "Power electronics and motion control".
 Tutorial 1 of the international conference on IEEE-IECON'92, November 9-13, 1992,
 San Diego, California, USA, Marriot Mission Valley.
- [BOW90] S.R. BOWES, J.C. CLARE, R. PEARCE
 "Vector controlled drive simulation using circuit analysis packages".
 Conference record of IEEE-IAS, 1990, Seattle, pp 495-501.
- [BRA89] S. BRASSARD
 "Méthodologie et modélisation floues des connaissances dans l'activité de conception en
 électrotechnique".
 Thèse INPG 1989.
- [BUL88] H. BULHER
 "Conception des systèmes automatiques".
 Presses Polytechniques Romandes, complément au traité d'électricité, 1988.
- [CAT92] S. CATELLANI
 "Etude sur la détection et la localisation de défauts en temps réel dans les éléments de
 puissance d'une association convertisseur-machine".
 Thèse INPG 1992.
- [CHE92] D. CHEN, B.K. BOSE
 "Expert system based automated selection of industrial AC drive".
 Conference record of IEEE-IAS, 1992, Houston, Texas, pp 387-392.
- [CHH92] S.M. CHHAYA, B.K. BOSE
 "Expert system based automated design technique of a voltage-fed inverter for induction
 motor drive".
 Conference record of IEEE-IAS, 1992, Houston, Texas, pp 387-392.

- [DOE88] R. LE DOEUFF, V. RAMAROMISA, S. HADDAD
 "GASPE : Générateur automatique de programme de simulation de procédés électromécaniques".
 Journée GRECO du 21 Avril 1988, Cachan, sur les Dispositifs et Systèmes Electriques.
- [DOE90] R. LE DOEUFF, F. SIMONOT, B. DAGUES, M. GRANDPIERRE, J. FAUCHER, HADDAD
 "La simulation numérique des ensembles convertisseur-machine-commande - Problèmes généraux - Les générateurs GASPE et SOSIE".
 Journées d'Etudes Nationales, thème "Commande", Lyon, 14-15 Juin 1990.
- [ENJ89] G. ENJALBERT
 "Développement d'un outil de conception assistée par ordinateur : le logiciel CIRCUIT".
 Thèse INPG 1989.
- [FAR91] P.S. FARAH
 "Simulation de commandes numériques sur le logiciel MC2".
 Rapport de projet de fin d'étude 3ème année ENSIEG-INPG, Juin 1991.
- [FAU89] J. FAUCHER, B. DAGUES, M. GRANDPIERRE
 "Générateur automatique de simulation. La méthodologie SOSIE. Application à la simulation de convertisseur-machine-commande".
 Journées SEE à caractère Industriel et Scientifique, C.A.O. et Simulation, Grenoble, 3 Mars 1989.
- [FAU91] J. FAUCHER, M. GRANDPIERRE
 "Quelques aspects de l'étude de la commande des machines électriques et des convertisseurs statiques à partir des schémas informationnels".
 Journées du Club EEA, Toulouse, 14-15 Mars 1991.
- [FAU92] J. FAUCHER, M. GRANDPIERRE
 "Les graphes informationnels de causalité. Application à la simulation des systèmes électriques".
 Journées du Club EEA Electrotechnique 92, 26-27 Mars 1992, au CNAM (Paris).

- [FOU87] C. FOULARD, S. GENTIL, J.P. SANDRAZ
 "Commande et régulation par ordinateur numérique de la théorie aux applications".
 Eyrolles 1987.
- [FRA91] F. FRANCOIS
 "Nombres flous, calcul flou".
 Rapport de DEA de Génie Electrique INPG-LEG, Juin 1991.
- [GEN91] A. GENTILHOMME
 "COCASE un système expert d'aide à la conception d'appareillages électriques".
 Thèse INPG 1991.
- [GER90] L. GERBAUD
 "Simulation d'ensembles machine-convertisseur-commande : détection de défauts, amélioration des logiciels de simulation".
 Rapport de DEA de Génie Electrique INPG-LEG, Juin 1990.
- [GER92] L. GERBAUD, J. BIGEON, G. CHAMPENOIS
 "Modular approach to describe electromechanical systems. Using Macsyma to generate global approach simulation software".
 Proceeding of IEEE-PESC'92, Toledo, Spain, June 29 - July 3, 1992.
- [GIL75] J.C. GILLE, M. CLIQUE
 "La représentation d'état pour les systèmes dynamiques".
 Eyrolles, 1975.
- [HAT91] J.P. HATON, N. BOUZID, F. CHARPILLET, M. HATON, B. LAASRI, H. LAASRI, P. MARQUIS, T. MONDOT, A. NAPOLI
 "Le raisonnement en intelligence artificielle. Modèles, techniques et architectures pour les systèmes à bases de connaissances".
 Inter-Editions, Paris 1991. iia : informatique - intelligence - artificielle
- [HAU] J.P. HAUTIER, G. MANESSE
 "Utilisation des réseaux de Pétri pour l'analyse des systèmes électrotechniques".
 Techniques de l'Ingénieur, traité Génie Electrique, D 3740-1 à D 3740-20.

- [HSI92] J. HSIEH, C.C.LIU
 "An integrated knowledge and algorithm based method for power converter design".
 Proceedings of IEEE-PESC'92, Toledo, Spain, June 29-July 3, 1992, pp 1316-1323.
- [INP92A] "Journées de formation Télémécanique" réalisées dans le cadre de la formation
 continue de l'INPG, 1991-1992.
- [INP92B] "Cours d'électrotechnique donnés à l'ENSIEG-INPG"
- [IUN] C. IUNG, B. TRIBOUT, R. LE DOEUFF
 "Simulation numérique des systèmes électromécaniques complexes".
 Rapport.
- [JAU89] D. JAUME, S. CHELLIEZ, M. VERGE
 "Applications du formalisme d'état à la commande des systèmes continus".
 Eyrolles 1989.
- [JOU91] D. JOUVE
 "Commande numérique autoréglable d'un actionneur synchrone".
 Thèse INPG 1991.
- [KAU87] A. KAUFMANN
 "Nouvelles logiques pour l'Intelligence Artificielle".
 Ed. Hermes, 1987, traité de nouvelles technologies, série mathématiques appliquées.
- [LIS87] "Le-Lisp de l'INRIA".
 Version 15-21, manuel de référence, Décembre 1987.
- [MAC88] "Macsyma reference manual version 13."
 Computer aide Mathematics Group of Symbolics, Inc, Nov.1988.
- [MAN88] G. MANESSE
 "Méthode de description fonctionnelle des convertisseurs et de leur commande".
 Journée GRECO du 21 Avril 1988, Cachan.
- [MEY91] B. MEYER
 "Conception et programmation par objets".
 Inter-Editions, Pars 1990. iia : informatique - intelligence - artificielle

- [MID76] R.D. MIDDLEBROOCK, S. CUK
 "A general unified approach to modelling switching converter power stages".
 Conference record of IEEE PESC'76, June 8-10, 1976, Cleveland, OH, pp 18-34.
- [MOH89] M. MOHAN, T.M. UNDERLAND, W.P. ROBBINS
 "Powers electronics : converters, applications and design".
 Ed. John Wiley & Sons, 1989.
- [NOR92] O. NORMAND
 "Conception d'un outil général de simulation des systèmes de conversion d'énergie électrique et de leur commande".
 Thèse INPG 1992.
- [PAZ92] F. PAZOS FLORES
 "Machine synchrone autopilotée : modélisation pour la simulation et la commande. Eléments d'optimisation du pilotage".
 Thèse INPG 1992.
- [PET92] A. PETIT
 "Conception et réalisation d'un exécutif temps réel dédié à la commande de systèmes actionneurs électriques".
 Thèse INPG 1992.
- [PIN] D. PINCHON
 "Langages de calcul formel".
 Techniques de l'Ingénieur, H 2340-1, H 2340-12
- [PRA91] G. PRADEL, R. VIVES-FOS, S. ALLANO
 "Systèmes d'aide au choix et à la conception des électrobroches et des moteurs d'axes de machines outils d'usinage à très grande vitesse".
 Eleventh International Conference, Expert systems & their applications, Specialized Conference : Artificial Intelligence & Electrical Engineering, Avignon, May 27-31, 1991, pp 95-104.
- [SAH91] O. SAHRAOUI
 "Méthodologie de modélisation des convertisseurs statiques en vue de la simulation de systèmes électromécaniques complets".
 Rapport de DEA de Génie Electrique INPG-LEG, Juin 1991.

- [SEG85] G. SEGUIER, F. NOTELET
"Electrotechnique industrielle".
Technique et Documentation, Lavoisier, 1985.
- [SIM90] F. SIMONOT, R. LE DOEUFF, S. HADDAD, M.F. BENKHORIS
"Générateur automatique de programmes de simulation : le système GASPE".
Revue Générale de l'Electricité, n°1, Janvier 1990, pp 5-9.
- [SME92] "SMECI : Manuel de référence, version 1.65"
ILOG S.A., Mai 1992.
- [TRI91] F. TRICHON
"Modélisation du processus de conception des machines électriques : le système expert
DAMOCLES".
Thèse INPG 1991.
- [VOG89] VOGEL
"Génie cognitif".
Ed. Masson, 1989.

A U T O R I S A T I O N de S O U T E N A N C E

VU les dispositions de l'Arrêté du 30 mars 1992 relatif aux Etudes doctorales

VU les rapports de présentation de

- . Monsieur le Professeur GIAMBIASI Norbert
- . Monsieur le Professeur SOL Charles

Monsieur GERBAUD Laurent

est autorisé(e) à présenter une thèse en soutenance en vue de l'obtention du diplôme
de DOCTEUR de L'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE, spécialité
" Génie Electrique "

Fait à Grenoble, le 8 avril 1993

/ BV.

Pour le Président de l'INPG
et par délégation
le Directeur de l'Ecole Doctorale
J.L. LACOUME



RESUME

L'objectif de cette thèse est de définir un outil d'aide à la conception des associations machine - convertisseur et de leurs commandes à partir d'un cahier des charges d'une application donnée. Afin d'établir les performances de ces ensembles, le concepteur utilise dans un premier temps, un outil de simulation de ces associations avant de construire un prototype. Les premiers chapitres sont consacrés à la description d'un tel outil avec comme objectif principal de permettre une génération automatique d'équations globales et de programmes de simulation à partir d'une description modulaire de l'application.

Les derniers chapitres abordent les principes d'un système expert d'aide à la conception de ces associations. Ils présentent le côté itératif et systématique de la conception et proposent un formalisme de description de la connaissance qui a permis de mettre en oeuvre un système expert où certaines étapes du processus de conception sont systématisés.

ABSTRACT

The aim of this work is to define a CAD package of drives from the specifications of a given application.

To obtain the drive performances, the designer uses simulation package before building a prototype. The second to fourth chapters present such a package whose main purpose is to automatically generate global equations and simulation softwares.

The last two chapters give the principle of an expert system to aid the design of drives. They present the iterative and systematic aspects of the design process and they propose a formalism to describe the knowledge. This has allowed us to build an expert system that automates some steps of this process.

Mots clés : association machine-convertisseur-commande, méthodologie, simulation numérique, générateur d'équations et de programmes de simulation, calcul formel, conception, système expert, règle de production, formalisme objet.

