THESE présentée
pour l'obtention
du
DIPLOME de DOCTEUR de 3e CYCLE
à
L'UNIVERSITE DE PARIS VI

spécialité : Mathématiques Appliquées

mention : Informatique

par Madame Monique GRANDBASTIEN

Sujet de la thèse : Un programme qui résoud formellement des équations trigonométriques par des procédés heuristiques.

Soutenue le juin 1974 devant la Commission composée de :

Président	M. J.C. SIMON
Examinateurs	M. J. PITRAT
	Mme F. MADAULE
Invité	M. C. PAIR

THESE présentée
pour l'obtention
du
DIPLOME de DOCTEUR de 3e CYCLE
à
L'UNIVERSITE DE PARIS VI

spécialité : Mathématiques Appliquées

mention : Informatique

par Madame Monique GRANDBASTIEN

Sujet de la thèse : Un programme qui résoud formellement des équations trigonométriques par des procédés heuristiques.

Soutenue le juin 1974 devant la Commission composée de :

Président	M. J.C. SIMON
Examinateurs	M. J. PITRAT
	Mme F. MADAULE
Invité	M. C. PAIR

Je tiens à remercier :

Monsieur le Professeur J.C. SIMON qui me fait l'honneur de présider ce jury ;

Monsieur J. PITRAT, Maître de Recherches au C. N. R. S., qui m'a orientée vers cette recherche et m'a prodigué conseils et encouragements tout au long de sa réalisation ;

Madame F. MADAULE, Professeur, qui a bien voulu s'intéresser à ce travail et participer au jury ;

Monsieur C. PAIR, Professeur, qui m'a accueillie à NANCY et a accepté de se joindre à ce jury.

Je tiens à remercier également Madame ZANNAD qui a assuré avec soin la réalisation matérielle de cette thèse ainsi que le service de reprographie de l'I. R. E. M. de NANCY.

UN PROGRAMME QUI RESOUD
FORMELLEMENT DES EQUATIONS TRIGONOMETRIQUES
PAR DES PROCEDES HEURISTIQUES

INTRODUCTION

Le but du travail que nous présentons ici était de concevoir et réaliser un programme heuristique de résolution d'équations trigonométriques.

Il nous a semblé important de situer ce travail dans le cadre des recherches actuelles en Intelligence Artificielle et de commenter un certain nombre de réalisations récentes dont certains aspects se rapprochent de la nôtre. C'est l'objet des deux premières parties de cette thèse.

Dans une troisième partie, nous expliquons comment nous avons élaboré la méthode de résolution qu'utilise le programme et nous décrivons précisément le fonctionnement de ce programme. Des exemples de résolution sont détaillés.

La dernière partie est consacrée aux problèmes techniques de représentation d'expressions, calcul formel, simplifications et aux résultats obtenus par le programme.

Nous indiquons en conclusion comment cette étude pourrait être prolongée.

1ère partie

L'INTELLIGENCE ARTIFICIELLE :
DES APPLICATIONS NOUVELLES POUR LES ORDINATEURS

I - BUTS DE L'INTELLIGENCE ARTIFICIELLE

L'apparition des ordinateurs puissants dont nous disposons actuellement a souvent éveillé dans le public des sentiments d'admiration devant leur rapidité et leur fiabilité. Cependant, ceux qui ont étudié d'un peu plus près leur fonctionnement et les possibilités d'utilisation qu'ils offraient se sont rendus compte qu'on leur demandait presque exclusivement des opérations de nature algorithmique et très répétitives.

Beaucoup d'applications classiques (calculs numériques, paies, gestion de stocks, etc...) sont maintenant au point et nombreux sont ceux qui souhaitent utiliser les machines pour répondre à des besoins nouveaux. L'une des idées directrices pour ces recherches nouvelles a été d'essayer d'obtenir pour l'accomplissement de certaines tâches un "comportement intelligent" de la part de l'ordinateur, ceci en faisant référence au comportement humain dans une situation analogue.

Pour accomplir une action, l'homme dispose de deux types de facultés ; les unes lui ont permis et lui permettent de percevoir un certain nombre d'informations extérieures à lui-même, les autres lui permettent de raisonner et de décider. Les auteurs d'un récent article (27) parlent "d'intelligence inductive" pour la faculté de reconnaître des informations fournies par les sens et "d'intelligence déductive" pour la capacité de construire des raisonnements et d'engendrer des actions. Les recherches en reconnaissance automatique des formes ont pour but de simuler l'intelligence inductive de l'homme et l'intelligence artificielle veut simuler son intelligence déductive.

I - DOMAINES D'APPLICATION ET DE RECHERCHES ACTUELS

Avec des buts aussi vastes, les domaines de recherches et d'applications ne manquent pas à l'intelligence artificielle. Cependant, à quelques exceptions près, tous les travaux actuels concernent des applications linguistiques ou des résolutions de problèmes ; ils font souvent appel à la perception de tout ou partie du

monde extérieur, c'est-à-dire à des procédures de reconnaissance des formes.

Dans les applications linguistiques, on peut inclure des recherches sur la traduction automatique d'une langue naturelle dans une autre, sur la documentation automatique, les systèmes de questions-réponses et les conversations en langue naturelle avec un ordinateur dont le programme de WINOGRAD (67) est un bel exemple.

La résolution de problèmes touche à des domaines très divers puisqu'on définit un problème par un couple (but, moyens). Le but, c'est l'objectif à atteindre, les moyens, ce sont les règles permises pour atteindre le but à partir de la situation initiale.

Premier exemple :
Jouer au bridge. Le but est d'essayer de gagner... ou de perdre honorablement la partie ; les règles indiquent les cartes que l'on a le droit de jouer en fonction des situations.

Deuxième exemple :
Démontrer une conjecture dans une théorie mathématique. Le but est de prouver que la conjecture est un théorème, les moyens sont les axiomes et les théorèmes déjà connus de cette théorie.

Troisième exemple :
Pour un robot, transporter un objet d'un point A à un point B dans un univers donné. Le but est de réaliser l'action proposée, les moyens sont les actions élémentaires que le robot peut accomplir.

Ces exemples sont empruntés aux trois principales catégories de travaux existants : les jeux, la démonstration de théorèmes et les robots.

Il faut noter qu'une grande partie des efforts accomplis en ce moment en intelligence artificielle portent sur des problèmes que les gens trouvent très simples mais qui sont difficiles à automatiser, en particulier la perception et le langage. On a de bons programmes pour jouer aux échecs ou faire du calcul intégral du niveau Mathématiques Spéciales mais on ne sait pas reconnaître un objet un peu compliqué ou un individu. Les notions traditionnelles de "facilité" et de "difficulté" sont remises en cause, d'autres critères interviennent, par exemple

la description de l'environnement dans lequel doit être réalisée une action est souvent bien plus complexe que l'action elle-même.

Ces critères différents appellent des méthodes et des outils originaux.

III - CARACTERES ET METHODES

Pour réaliser un programme doué d'intelligence, une première base de travail est l'observation et l'imitation du processus intelligent humain dans tous les domaines où il intervient.

Les recherches actuelles menées en neurophysiologie et en psychologie nous apporteront sans doute des renseignements intéressants car les mécanismes physiologiques mis en jeu par l'intelligence humaine sont très mal connus. Mais comme le disait récemment un chercheur américain, "étudier la structure des plumes n'est pas une bonne méthode pour savoir comment vole un oiseau et en intelligence artificielle, il ne suffit pas d'étudier les neurones du cerveau pour savoir comment fait l'homme pour penser".

Cette observation apportera donc des éléments pour construire le programme, mais le but de l'intelligence artificielle n'est pas de simuler le comportement humain ; on parlera de programme "intelligent" au sens des résultats qu'il obtient dans un domaine donné, même si la méthode utilisée ne nous paraît pas astucieuse. Un programme résulte de l'accumulation d'un certain nombre de sous-programmes qui effectuent des actions réduites dans des domaines restreints ; rien ne prouve que l'intelligence humaine générale résulte d'une pareille accumulation. Il utilise un ordinateur qui a une structure et des performances données, il s'agit de bien utiliser cette structure particulière et ses capacités (rapidité et fiabilité en particulier), tout comme nous utilisons au mieux notre cerveau.

Les méthodes d'un programme ne seront donc pas forcèment les nôtres, elles dépendent beaucoup du domaine dans lequel on veut travailler ; aussi, nous nous bornerons à donner au chapitre suivant un aperçu de celles qu'on rencontre en résolution de problèmes.

IV - PERSPECTIVES

Une bonne partie des programmes actuels résolvent des problèmes très particuliers. C'est probablement une étape indispensable dans la recherche de la généralité ; certains se sont déjà lancés dans cette voie (20), (38),... , elle est difficile et ambitieuse mais capitale pour le développement des applications de l'intelligence artificielle.

Certaines études particulières font parfois sourire. A quoi cela peut-il servir d'écrire un programme pour jouer aux échecs ? L'intelligence artificielle aura-t-elle un jour des applications précises et utiles ?

Les programmes écrits sur des sujets "gratuits" servent souvent à mettre au point des méthodes qui se révèlent efficaces dans d'autres domaines et les "retombées" de ces recherches ont été et seront encore utiles à d'autres disciplines. Les systèmes de "questions-réponses" ou d'aide au calcul formel représentent les applications d'aujourd'hui. Celles de demain sont évidemment moins précises ; il existe deux vastes projets de robots à l'étude dans des laboratoires aux U. S. A. L'un voudrait construire un robot pour explorer des planètes lointaines, il pourrait prendre seul les décisions qu'imposeraient les circonstances. L'autre robot devrait remplacer les ouvriers dans les travaux d'une chaîne de construction automobile.

Ces perspectives font parfois rêver ! Si elles ne sont pas encore du domaine des réalités, elles ont quitté celui de la science-fiction. Elles ouvriront des voies nouvelles au développement de l'informatique traditionnellement appliquée au calcul numérique et à la gestion et n'ont pas fini de nous étonner.

2ème partie

LA TRIGONOMETRIE : UN ENVIRONNEMENT INTERESSANT POUR ETUDIER LA RESOLUTION DE PROBLEMES

Nous définissons d'abord ce que l'on entend par résolution de problèmes en Intelligence Artificielle, puis nous étudions des travaux dans le cadre de la trigonométrie.

I - LES APPROCHES POSSIBLES EN RESOLUTION DE PROBLEMES

Nous avons vu dans le chapitre précédent qu'on pouvait aborder des problèmes très variés. On peut distinguer trois types d'approches de ces problèmes :

1) - Le combinatoire :

C'est une idée très naturelle. Elle consiste à essayer à chaque étape toutes les règles permises et ceci jusqu'à ce qu'on atteigne la solution recherchée. Malheureusement, si a règles sont applicables à la situation initiale, au bout de n étapes, on aura a^n situations possibles. Cette croissance exponentielle du nombre de situations rend la méthode souvent inapplicable. On ne saura jamais si le programme s'arrête parce qu'il ne dispose plus de temps machine ou d'espace mémoire ou simplement parce qu'il n'y a pas de solution.

Cependant, l'idée n'est pas à rejeter totalement. Les récents travaux de SIKLOSSY et MARINOV (41), (42), ont montré qu'on avait parfois intérêt à insérer des procédures combinatoires limitées pour la résolution de certains problèmes.

On peut aussi essayer de limiter la croissance exponentielle de l'arborescence des situations engendrées en écartant les noeuds qui ne conduisent certainement pas à la solution.

2) - Un algorithme :

On cherche pour certains problèmes une méthode qui conduit au résultat s'il en existe ou même mieux, une méthode qui indique le résultat s'il existe et qui indique que la solution n'existe pas dans le cas contraire. On connaît des algorithmes de ce type pour certains jeux ou pour les théories

mathématiques où il existe une procédure de décision.

Une formulation mathématique qui convient théoriquement à de nombreux problèmes est celle du calcul des prédicats. La méthode connue sous le nom de "résolution" (40) est la suivante :

- représenter le domaine considéré par un ensemble de clauses dans le calcul des prédicats de premier ordre, sous forme normale disjonctive :
  (A ou B ou non (C))

- représenter ainsi la partie hypothèse de la conjecture à démontrer et la négation de la conclusion.

- utiliser la règle de déduction :
  si (A ou non (B)) et (B ou C) sont vraies, alors on peut en déduire (A ou C)
  et essayer de prouver que la négation de la conclusion est en contradiction avec l'hypothèse.

La méthode est évidemment séduisante ; elle revient simplement à démontrer un théorème du calcul des prédicats. Elle a donné de bons résultats dans certains domaines, bien qu'il n'existe pas de procédure de décision pour la calcul des prédicats du premier ordre.
Cependant, on peut lui faire deux reproches :

. Certains problèmes se prêtent très mal à une formulation dans le calcul des prédicats, en particulier ceux de la vie courante pour lesquels on aboutit à des énoncés très longs.

. Lorsqu'une démonstration est obtenue, on peut difficilement en suivre le déroulement.

Il y a donc des catégories de problèmes pour lesquelles il faut utiliser d'autres méthodes.

3) - Les heuristiques :

Nous utilisons pour calculer, pour jouer et pour réagir devant certaines situations quotidiennes, des "pseudo-règles" qui nous sont si familières

que nous ne savons plus d'où elles nous viennent.

Nous arrivons à lire des manuscrits mal calligraphiés où les lettres n'ont que rarement leur forme "classique", comment faisons-nous ? On nous a appris à jouer atout à la belote, à jouer dans la forte du mort au bridge, si l'on est avant lui, pourquoi ? Pour résoudre un problème de géométrie nous faisons une figure qui suggère un raisonnement. Ces règles, nous les appliquons souvent et nous savons pourtant que dans certains cas nous pouvons nous en passer ou même qu'il ne faut pas les utiliser.
Ce sont des heuristiques, c'est-à-dire des moyens qui aident à trouver rapidement la solution mais qui peuvent momentanèment ou définitivement nous en écarter.

La justification de l'emploi d'une heuristique est essentiellement statistique, l'heuristique est d'autant meilleure qu'elle conduit souvent et rapidement à la solution.

Beaucoup de programmes d'intelligence artificielle utilisent des heuristiques. Elles peuvent être de natures très diverses suivant les buts que l'on a assignés au programme. Pour travailler dans un domaine particulier, on utilisera les caractères et règles propres à ce domaine. Nous en verrons de nombreux exemples en trigonométrie dans les chapitres suivants. Si au contraire, on souhaite un programme général, il faut des heuristiques qui s'appliquent à toute une famille de problèmes ; par exemple pour un programme général de jeux, on peut avoir : "diminuer la mobilité potentielle de l'adversaire", pour des résolutions de problèmes, le G.P.S. décrit au paragraphe suivant a des heuristiques très générales.

Pour un problème particulier, le programmeur devra donc rechercher une liste de "règles" et/ou d'heuristiques fines. Pour un problème général, il lui faudra soit donner des heuristiques très générales, soit rendre le programme capable de trouver lui-même de bonnes heuristiques. Une fois cette liste établie, les choix heuristiques interviennent encore à deux niveaux dans un programme : pour déterminer à un instant donné l'ordre dans lequel ces règles doivent être appliquées et pour déterminer l'intérêt qu'on a à utiliser une règle reconnue applicable.
Nous allons voir au paragraphe suivant des exemples de programmes heuristiques.

II - QUELQUES RESOLUTIONS DE PROBLEMES FAISANT APPEL AU CALCUL TRIGONOMETRIQUE

Plusieurs programmes ont déjà été écrits en intelligence artificielle pour résoudre des problèmes de trigonométrie. C'est en effet un domaine où les règles sont bien définies et qui est pourtant complexe : un élève de classe terminale inexpérimenté en calcul trigonométrique peut mettre beaucoup de temps pour démontrer des identités apparemment simples. Il est surtout très dangereux dans ce type de calcul d'appliquer sans discernement les formules dont on dispose : l'expérience prouve en effet qu'on arrive rapidement à des expressions trop compliquées pour pouvoir être utilisées ou retransformées efficacement.

L'objet de ce paragraphe, après avoir rappelé la nature du calcul trigonométrique, est de présenter les principales réalisations connues dans ce domaine, d'en tirer certains enseignements ou d'en souligner les limites.

Il semble que nous puissions dégager trois types de programmes travaillant dans un environnement trigonométrique :

. Les programmes qui se veulent généraux, c'est-à-dire applicables à des domaines très divers et qui fonctionnent en particulier dans un environnement trigonométrique.

. Les programmes plus spécifiques et plus performants, qui ont tous pour objet de démontrer des identités trigonométriques.

. Les programmes qui manipulent des expressions trigonométriques et qui ont, par conséquent, à faire intervenir certaines heuristiques propres à ce type de calcul, sans que ce soit leur but principal.

Nous verrons en conclusion quels problèmes restent à résoudre.

1) - Les problèmes de trigonométrie :

La trigonométrie, étymologiquement "mesure des triangles", c'est-à-dire évaluation de leurs côtés et de leurs angles, a conduit les géomètres à attacher à un angle des nombres qui le caractérisent. Ces nombres appelés "lignes trigonométriques" ont permis de résoudre de nombreux problèmes de géométrie plane mais se sont révélés très précieux dans la résolution de beaucoup d'autres questions. C'est pourquoi on est amené à les étudier pour eux-mêmes aussi bien qu'à travers leurs applications.

En dehors de tout contexte, on a cherché à établir des relations entre une ou plusieurs lignes trigonométriques d'un ou plusieurs arcs. Ces travaux ont permis d'établir les "formules" classiques bien connues, d'autres problèmes sont souvent proposés pour établir des relations plus particulières, ils sont connus sous le nom de démonstration d'identités trigonométriques.

En géométrie plane, les problèmes les plus classiques sont ceux de la résolution des triangles, c'est-à-dire du calcul des côtés et des angles à partir de certaines données et leurs applications.

Mais les lignes trigonométriques d'un arc interviennent dans beaucoup de résultats partiels et on les rencontre donc dans les résolutions d'équations, le calcul de limites, le calcul intégral, etc...

Dans tous ces domaines, on est amené à modifier la forme des expressions en utilisant les propriétés des lignes trigonométriques qu'elles contiennent, c'est-à-dire en utilisant au mieux la classique "panoplie" de formules dont on dispose. Le problème ainsi posé est de nature heuristique, nous savons par expérience qu'il ne faut pas développer n'importe quel terme si l'on veut continuer à maîtriser les calculs que l'on effectue sur une expression. Il faudra donc définir des critères pour décider des transformations souhaitables en fonction du but recherché. C'est ce qui a été fait dans les programmes que nous décrivons.

2) - Les programmes généraux qui peuvent résoudre des problèmes de trigonométrie :

. Le "General Problem Solver" :

Le premier programme général mis au point en Intelligence Artificielle est le General Problem Solving Program (G.P.S.) de NEWELL SHAW et SIMON. Il date des années 1959 et il est intéressant de noter que ses auteurs le présentent à la fois comme un modèle de simulation d'intelligence en psychologie et comme un programme d'intelligence articifielle capable de résoudre certains problèmes non triviaux et très divers.

L'aspect "étude du comportement humain au cours de la résolution d'un problème" les a amenés à dégager d'une part les 3 notions de situation initiale, situation finale ou but à atteindre et règles permises qui peuvent caractériser un problème, d'autre part les notions de différences entre situation initiale ou intermédiaire et situation finale et de possibilités pour les règles de réduire certaines différences.

L'aspect "écriture d'un programme général d'intelligence artificielle" a conduit à formaliser séparèment la méthode générale "mean-end analysis" en analyse du but et des moyens et la description du problème posé et de son environnement "task environment".

. La démarche du G.P.S. :

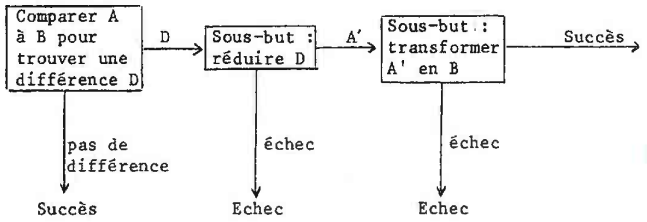
G.P.S. travaille dans un environnement formé d'objets qui peuvent être transformés en d'autres objets par un certain nombre d'opérateurs. Il détecte des différences entre ces objets et se donne successivement plusieurs buts qui s'ils sont atteints, conduisent à la solution du problème posé. Il y a trois types de buts :

- Transformer un objet A en un objet B
- Réduire une différence D entre les objets A et B
- Appliquer l'opérateur Q à l'objet A.

Pour atteindre ces buts, G.P.S. emploie principalement trois méthodes résumées par les schémas suivants :

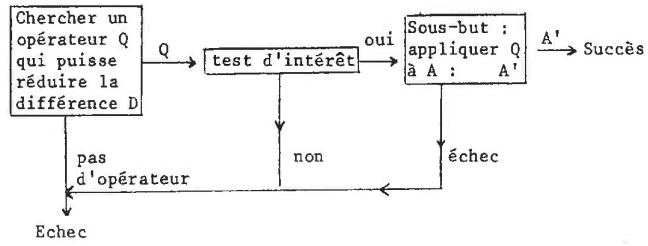
But : transformer un objet A en un objet B

Méthode :



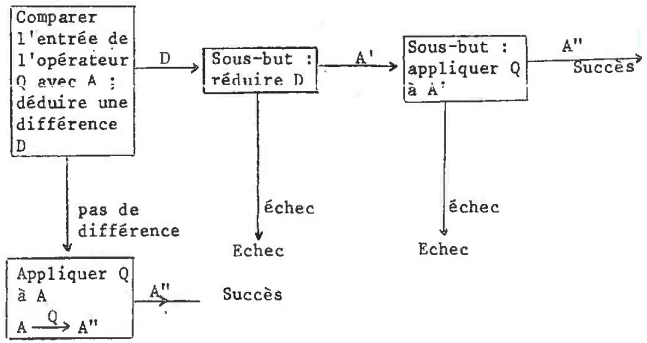
But : réduire la différence D entre un objet A et un objet B

Méthode :



But : appliquer l'opérateur Q à l'objet A

Méthode :



Ces méthodes forment un système récursif qui génère un arbre de sous-buts pour résoudre le problème donné. Chaque fois que le programme rencontre une difficulté, il créé un nouveau but intermédiaire pour la surmonter. G.P.S. possède en outre des tests pour éviter une expansion démesurée de l'arbre généré :

Le principal d'entre eux est appliqué lorsqu'il détecte une différence et engendre un but pour réduire cette différence. Les différences sont classées par ordre de difficulté décroissante. G.P.S. essaie de réduire d'abord les différences les plus "difficiles".

Les autres tests concernent des limites pour la profondeur totale de l'arbre des buts et des comparaisons pour ne pas générer des buts figurant déjà dans l'arbre sur les mêmes objets.

. Le fonctionnement en trigonomètrie :

Description du domaine de travail.
Les objets : ce sont les expressions algébriques et trigonomètriques ordinaires.
Exemple : $\sin^2 x - \cos^2 x$, $1 + tgx$, 1....

- Les opérateurs :

AO : (Réductions algébriques élémentaires appliquées récursivement à partie des expressions les plus internes)

$A + (B + C) \longrightarrow A + B + C$
$A . (B . C) \longrightarrow A . B . C$
$A + 0 \longrightarrow A$
$A + A \longrightarrow 2 . A$
$A - A \longrightarrow 0$
$A . 0 \longrightarrow 0$
$A . 1 \longrightarrow A$
$A . A \longrightarrow A^2$
$A^B . A^C \longrightarrow A^{B+C}$
$A^0 \longrightarrow 1$
$0^A \longrightarrow 0$
$A^1 \longrightarrow A$
$(A^B)^C \longrightarrow A^{B . C}$

A1 : $(A - B)(A + B) \longrightarrow A^2 - B^2$

A2 : $(A + B)^2 \longrightarrow A^2 + 2AB + B^2$

A3 : $A (B + C) \longrightarrow A . B + A . C$

T1 : $tg\ x \longrightarrow 1/cotg\ x$

T2 : $tg\ x . cotg\ x \longrightarrow 1$

T3 : $tg\ x \longrightarrow \sin x/\cos x$

T4 : $cotg\ x \longrightarrow \cos x/\sin x$

T5 : $\sin^2 x + \cos^2 x \longrightarrow 1$

- Les différences entre les objets :

Ces différences s'appliquent aussi bien aux expressions dans leur ensemble qu'aux arguments des fonctions trigonométriques, par exemple :

△V : Une variable apparaît dans un membre de l'identité et pas dans l'autre.
Exemple : $tg\ x + cotg\ x$ diffère par - V de $tg\ x$ puisqu'il faut "enlever" $cotg\ x$ pour passer de l'une à l'autre.
$\sin a$ diffère par + V de $\sin (a + b)$.

△N : Une variable n'apparaît pas le même nombre de fois dans les deux membres.

△T : Il y a une différence de signe entre les 2 expressions.
Exemple : $tg\ x + cotg\ x$ et $- (tg\ x + cotg\ x)$

△C : Il y a une différence de connectives entre les 2 objets :
Exemple : $tg\ x + cotg\ x$ et $tg\ x . cotg\ x$.

- La table de connexion entre les différences et les opérateurs :

	AO	A1	A2	A3	T1	T2	T3	T4	T5
△V	-				t	t	t	t	t
△C	+	+	+	+					
△N	-	+	+	+	t	t	t	t	t
△T	+								

Un plus ou un moins dans une case de cette table signifie que l'opérateur situé dans la colonne influe sur la différence située sur la ligne correspondante.

Un t signifie que l'opérateur correspondant est accepté pour réduire la différence seulement s'il produit des fonctions qui interviennent déjà dans l'expression.
Par exemple, l'opérateur $tg\ x \longrightarrow \frac{\sin x}{\cos x}$ n'est pas accepté si toute l'expression est exprimée en tg x et cotg x.

- Hiérarchie des opérations :

Des différences dans une sous-expression sont signalées avec les différences sur l'expression elle-même.
Pour deux objets donnés, les différences sont rangées dans l'ordre décroissant suivant :

$$+ V, - V, \Delta C, + N, - N, \Delta T$$

- Exemple de résolution d'un problème :

Montrer que $(tg\ x + cotg\ x) \sin x \cos x = 1$.
Ceci se traduit pour G.P.S. par un premier but à atteindre :
transformer $(tg\ x + cotg\ x) \sin x \cos x$ en 1.
La figure suivante indique les différents sous-buts engendrés pour atteindre le résultat.

L1 = $(tg\ x + cotg\ x) \sin x \cos x$
L0 = 1

But 1 : Transformer L1 en L0.
Méthode : on cherche une différence entre L1 et L0
ordre : 1) une différence de variables
la première variable rencontrée est tg x

But 2 : Réduire cette différence
Méthode : appliquer A0 ou T1

But 3 : Appliquer A0 à L0 ---- ECHEC

But 4 : Appliquer T1 à L1 ---- OUI

But 5 : L2 = $((1/cotg\ x) + cotg\ x) \sin x \cos x$

But 6 : Transformer L2 en L0.
Etude des différences : variables ? ---- OUI

But 7 : Réduire la différence de variables
première variable rencontrée : cotg x

But 8 : Appliquer A0 à L2 ---- ECHEC

But 9 : Appliquer T4 à L2

But 10 : L3 = $(1/(\cos x/ \sin x) + \cos x/\sin x) \sin x \cos x$

But 11 : Transformer L3 en L0.
Etude des différences : variables ? ---- OUI

But 12 : Réduire cette différence
première variable rencontrée : cos x

But 13 : Appliquer A0 à L3.

But 14 : L4 = $\left(\frac{\sin x}{\cos x} + \frac{\cos x}{\sin x}\right) \sin x \cos x$

But 15 : Transformer L4 en L0
Etude des différences : variables ? ---- OUI

But 16 : Appliquer A0 à L4 ---- ECHEC

But 17 : Appliquer T5 à L4 ---- OUI

But 18 : Transformer L4 en C(T5)

But 19 : Réduire la différence de connective entre L4 et C(T5)

But 20 : Appliquer A0 à L4 ---- ECHEC

But 21 : Appliquer A1 à L4 ---- OUI

But 22 : Transformer L4 en C (A1)

But 23 : Réduire la différence de connectives entre L4 et C(A1) ---- ECHEC

But 24 : Appliquer A3 à L4

But 25 : Transformer L4 en C (A3)
L5 = $\frac{\sin x}{\cos x} \sin x \cos x + \frac{\cos x}{\sin x} \sin x \cos x$

But 26 : Transformer L5 en C(T5)
Différence ? oui

But 27 : Réduire la différence de connectives entre les parties gauches de L5 et C(T5).

But 28 : Appliquer A0 à L5

$L6 = \sin^2 x + \frac{\cos x}{\sin x} \sin x \cos x$

But 29 : Transformer L6 en C(T5).

But 30 : Réduire les différences de connectives entre les parties de L6 et de C(T5).

But 31 : Appliquer A0 à la partie droite de L6

$L7 = \sin^2 x + \cos^2 x$

But 32 : Transformer L7 en C(T5) ---- SUCCES

$L8 = 1$

But 33 : Transformer L7 en LOT

Succès général.

· Le programme DATAL :

Comme G.P.S., DATAL travaille dans un environnement composé d'objets et d'opérateurs ; il développe une arborescence d'objets intermédiaires et pour déterminer au cours de la résolution d'un problème le prochain objet à transformer ou l'opérateur à appliquer à tel objet, il opère à partir de "différences" reconnues entre les objets.

On peut trouver dans (19) la description de l'environnement que possède DATAL pour travailler en trigonométrie, ainsi que les démonstrations qu'il donne pour vérifier 18 identités trigonométriques.

Ces deux programmes généraux fonctionnent bien pour la vérification d'identités trigonométriques tant que ces identités ne comportent pas trop de termes. Dès que les démonstrations demandent des transformations locales sur un ou deux des termes d'une expression plus longue ou présentent des arguments complexes, l'arborescence devient très grande.
Il faudrait, si l'on voulait conserver la méthode, définir davantage de "caractéristiques" pour les objets, davantage de différences et établir une table de connexion très fouillée entre ces différences et des

opérateurs plus nombreux. C'est un peu ce qu'ont réalisé HOLDEN et JOHNSON avec des buts et un cadre différents.

3) - Les programmes de démonstrations d'identités trigonométriques :

. Programme de A.D.C. HOLDEN et D.L. JOHNSON (25) :

Le but de ce travail est de développer un programme de résolution de problèmes capable d'utiliser sa propre expérience pour améliorer ses performances et fondé sur des heuristiques assez générales. Cependant, contrairement aux programmes cités au paragraphe précédent, la méthode semble surtout conçue pour des calculs algébriques, trigonométriques ou logiques.

Heuristiques générales :
Elles sont au nombre de deux :

H1 : "Transformer en une forme "représentative" "
qui se décompose en trois heuristiques plus fines

H1a : Réduire le nombre de formes équivalentes en réarrangeant les symboles.

H1b : Transformer en une forme plus simple.

H1c : Transformer en une forme plus familière.

H2 : Appliquer une transformation qui possède certains des "caractères" de la situation présente si cette transformation doit produire une situation "souhaitable".

Application à la démonstration d'identités trigonométriques :

Pour le programme, une "situation" S est une identité à démontrer, c'est-à-dire une chaîne de symboles, les symboles pouvant être des constantes, des variables, des connectives (+, -, =...) et des fonctions trigonométriques.

Les règles utilisées pour réaliser H1a sont les suivantes :

. Décider d'un ordre pour l'apparition des différents types de symboles dans S (pour les connectives commutatives).
Par exemple, sur les fonctions trigonométriques, l'ordre choisi est :
sin x, cos x, tg x, cotg x, cosec x, sec x ;
cet ordre entraîne la transformation :
(sec x = cos x + tg x sin x) ⟶ (sin x tg x + cos x = sec x)

. Remplacer les connectives les moins usitées par d'autres :
le signe / est supprimé (sin x/cos x = tgx) ⟶ (sin x = cos x tg x)
le signe - " " ($\cos^2 x = 1-\sin^2 x$) ⟶ ($\sin^2 x + \cos^2 x = 1$)

Pour réaliser H1b on définit une forme plus simple comme une forme contenant moins de symboles :

. Appliquer la ou les transformations qui réduisent le plus le nombre total des symboles :
($\sin^3 x + \sin x \cos^2 x = \cos x \text{ tg } x$) ⟶ $\sin x = \cos x \text{ tg } x$)

. Supprimer des termes ou des facteurs dans les 2 membres :
($\sin x \text{ tg } x \cos x = \sin^2 x$) ⟶ (tg x cos x = sin x)
($\sin^2 x + \cos^2 x + \sin x = 1 + \sin x$) ⟶ ($\sin^2 x + \cos^2 x = 1$)

La notion de forme plus familière introduite au H1c fait appel au "score de familiarité" défini statistiquement pour chaque transformation utilisée. Elle est liée au fait que le programme tient compte de son expérience antérieure : s'il a souvent utilisé la règle sin tg x + cos x = sec x, il en déduira que sec x est une forme plus "familière" que sin x tg x + cos x.

L'heuristique H2 reprend de façon plus fine et plus précise une idée du G.P.S., à savoir appliquer un opérateur qui puisse rapprocher les caractères de la partie droite de l'idendité à vérifier de ceux de la partie gauche. Pour chaque situation, le programmeur établit 2 ensembles de caractèristiques - un pour le membre de droite
- un pour le membre de gauche.

Chaque transformation possède elle aussi un ensemble de caractéristiques pour chacun de ses membres. Il faut noter que ces ensembles sont relativement lourds :

pour $S = \sin^2 x + \sin^2 x \text{ tg}^2 x = \text{tg}^2 x$

on a $G = \{\sin x, \text{ tg } x, \sin^2 x, \sin x \text{ tg } x, \text{tg}^2 x + 1\}$

$D = \{\text{tgx}, \text{tg}^2 x\}$

A chaque caractéristique est en outre affecté un nombre mesurant sa complexité. On appliquera alors des opérateurs qui suppriment au moins une des caractéristiques présentes, à condition que sa complexité soit suffisante.
Une quarantaine d'exercices ont été présentés au programme et résolus de façon satisfaisante. Si l'on compare ses performances à celles de G.P.S. on constate que les identités proposées sont plus complexes, qu'il y a peu de tâtonnements. Cela vient du fait que tous les résultats intermédiaires sont conservés et surtout de la description très précise des identités que constituent les ensembles de caractéristiques.

A titre d'exemple, avec comme données

ID 1 : tg x = sin x/cos x
ID 2 : cotg x = cos x/sin x
ID 3 : cosec x = 1/sin x
ID 4 : sec x = 1/cos x
ID 5 : $\sin^2 x + \cos^2 x = 1$

Le programme démontre
$\sin x \text{ tg } x + \sin x \text{ tg}^3 x = \cos x + \cos x \text{ tg}^3 x = \sec^3 x$
en 10 étapes.

. <u>Une génération algorithmique de preuves pour des identités trigonométriques (JONCOUR (26))</u>

Nous allons voir ici une approche tout à fait différente du même problème : vérifier des identités trigonométriques.
L'auteur a abordé la question sous un angle d'abord plus théorique en se demandant s'il pouvait définir une classe $\mathcal{C}$ d'expressions trigonométriques pour laquelle le problème de l'égalité entre deux expressions était décidable. Une fois cette classe définie, il établit l'existence d'une forme canonique pour tout élément de cette classe.

Une méthode algorithmique de génération de preuves en découle naturellement :

Etant donnée une identité $\mathcal{E} \equiv o$ à vérifier,
vérifier que $\mathcal{E}$ appartient à la classe $\mathcal{C}$,
passer de la forme initiale de $\mathcal{E}$ à sa forme canonique,
conclure à l'identité ou à la non identité à 0.

L'avantage de la méthode est évident : en supposant que l'on ait assez de place ou de temps, on arrive toujours à montrer qu'une identité dont le 1er membre appartient à la classe $\mathcal{C}$ est ou n'est pas vraie.

Des identités complexes ont été vérifiées. L'auteur donne comme exemples significatifs les preuves de :

$$\cotg x + \tg x = (\operatorname{cosec}^2 x + \sec^2 x)/(\operatorname{cosec} x \cdot \sec x)$$

$$\cos x \cos (x + y) + \sin x \sin (x + y) = \cos y$$

$$\cos 3x = 4 \cos^3 x - 3 \cos x$$

$$\sin (x/2) = (1 - \cos x)/2$$

$$\cos 2x \cos 2y + \sin^2 (x - y) - \sin^2 (x + y) = \cos (2x + 2y)$$

$$\sin 5x = 5 \sin x - 20 \sin^3 x + 16 \sin^5 x$$

Sont évidemment exclues de cette preuve algorithmique les identités dont le 1er membre n'appartient pas à la classe $\mathcal{C}$.
On peut trouver une définition précise de $\mathcal{C}$ avec la notation de BACKUS dans (26), disons brièvement que la constante π en particulier n'est pas admise car des termes comme $\pi/6 = \frac{\sqrt{3}}{2}$ ou $\sin \frac{\pi}{4} = \frac{\sqrt{2}}{2}$ aménéraient sur la forme canonique des coefficients irrationnels dont on ne pourrait pas affirmer dans tous les cas qu'ils sont ou ne sont pas nuls.
Les auteurs suggèrent eux-mêmes une solution : considérer π comme une variable et autoriser certaines simplifications à conditions qu'elles n'introduisent pas de facteurs irrationnels. Ce n'est pas un trop gros inconvénient à notre avis.

L'inconvénient de la forme canonique définie est qu'elle est très lourde, les expressions sont entièrement développées, les arguments des fonctions trigonométriques ramenés à une variable simple. Il s'ensuit des temps de calcul longs dès qu'on manipule des expressions complexes et surtout des arguments complexes pour les fonctions trigonométriques. Pour les exemples 5 et 6, les temps d'exécution sur PDP 10 sont respectivement de 2 mn 15 et 8 mn 15. De plus, les démonstrations sont rarement courtes et élégantes. Mais ceci est plus une constatation qu'une critique, il faut juger le programme sur ses résultats qui sont bons. On peut seulement reprocher à cette méthode algorithmique sa lourdeur et sa lenteur.

4) - <u>D'autres programmes de résolution de problèmes dans les domaines algébriques et trigonométriques</u> :

Nous en citerons trois en insistant surtout sur la démarche générale de chacun des programmes.

. <u>Le Symbolic Automatic Integration (SAINT) de SLAGLE</u> (43), (44)

L'auteur s'est proposé de résoudre le problème de l'intégration formelle des fonctions d'une variable réelle. Il a écrit un programme pour I.B.M. 7090 en 1961 dont les performances sont celles d'un étudiant moyen de nos premiers cycles d'Université. A titre d'exemple, SAINT résoud

$$\int x e^{x^2} dx \longrightarrow \frac{1}{2} e^{x^2}$$

mais aussi

$$\int \frac{x^4}{(1 - x2)^{5/2}} dx \longrightarrow \arcsin x + \frac{1}{3} \tg^3 (\arcsin x) - \tg \arcsin x$$

SAINT travaille en générant une arborescence de buts intermédiaires.
Pour atteindre un but, il dispose de 3 types de transformations :

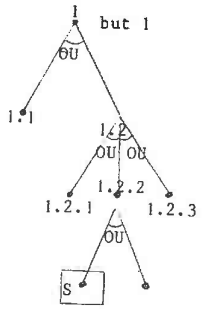
. Regarder si l'intégrale à calculer a l'une des 24 formes "standards", dans ce cas, il donne la solution

. Regarder si une des 8 transformations " presque algorithmiques" est applicable. Si oui l'appliquer, sinon :

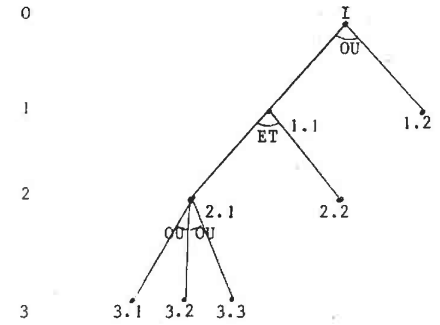
. Regarder si une transformation "heuristique" est possible.

Il faut remarquer que cette démarche est différente de celles que nous avons étudiées précédemment. Il n'y a pas cette fois de "caractères de but à atteindre" car on ne connaît pas la forme que prendra l'intégrale que l'on calcule, seuls peuvent être utilisés les caractères de la fonction à intégrer et des heuristiques, les deux types d'heuristiques retenus traduisent bien le comportement classique du mathématicien : des transformations quasi algorithmiques, c'est-à-dire que le mathématicien applique toujours s'il le peut et qui l'éloignent rarement de la solution, puis des transformations purement heuristiques dont le bien fondé est plus difficile à formaliser et qui en conséquence doivent être appliquées avec prudence.

Une autre différence de ce programme avec les précédents se situe au niveau de l'arbre des sous-buts qui comporte à la fois des noeuds ET et des noeuds OU. Dans les exemples précédents, à partir d'un noeud de l'arborescence, en lui appliquant plusieurs transformations, on générait de nouveaux buts, et il suffisait d'avoir une solution pour l'un d'entre eux ; on avait des noeuds OU.



SAINT peut décomposer le calcul d'une intégrale en plusieurs calculs d'intégrales et pour chaque calcul il peut appliquer plusieurs heuristiques, ce qui donne la structure suivante :



Pour calculer I, il faut avoir calculé :

soit l'intégrale figurant en 1.2

soit l'intégrale figurant en 1.1

Pour calculer 1.1, il faut avoir calculé

2.1 c'est-à-dire { soit 3.1 / soit 3.2 / soit 3.3 }

et

2.2

Ce schéma semble très utile dès qu'on a des problèmes complexes qu'on peut ramener à plusieurs problèmes apparemment plus simples. Sans être forcèment général, il semble adapté à la résolution de beaucoup de problèmes, nous en verrons d'autres exemples.

. SIN (Symbolic INtegration) de J. MOSES (32) :

En 1966-67, MOSES a repris l'idée de SLAGLE pour écrire un nouveau programme d'intégration formelle. Son approche est également celle d'une résolution de problèmes en Intelligence Artificielle. SLAGLE avait tente de reproduire à peu près le comportement du mathématicien. MOSES s'est assigné comme buts d'avoir un programme performant et un résultat utilisable par celui qui a posé le problème, c'est-à-dire du même "type" que la fonction à intégrer. Pour y arriver, il utilise des heuristiques assez puissantes, plus générales :

- essayer de résoudre le problème par une méthode générale et rapide :

  soit à calculer $\int f(x)\,dx$, mettre f(x) sous la forme $C.op\,(u(x)).u'(x)$

  où op est un opérateur élémentaire appartenant à une liste donnée
  c est une constante
  u une fonction de x.

- sinon, essayer l'une des 11 méthodes spécifiques à certaines classes d'intégrales(fonctions trigonométriques, exponentielles, radicaux)

- sinon, essayer une méthode "d'intégration par parties heuristique ", l'algorithme de RISCH.

MOSES note que les deux premières étapes sont très rapides et permettent de résoudre beaucoup de problèmes.

On peut dire, en conclusion, que SIN est du même type que SAINT mais plus performant par des heuristiques différentes.

. DALI (Démonstration Automatique de Limites) (28), (29)

Ce programme se propose de calculer la limite d'une expression f(x) lorsque x tend vers x_0 donné. Il a fonctionné sur plus d'une centaine d'exercices présentant des difficultés certaines. Il se décompose en deux parties POLLIM et TRANS :

- mettre sous forme de polynomes toutes les parties de f (x) qui s'y prêtent, faire toutes les opérations arithmétiques et développements limités possibles sur ces polynomes et essayer de conclure en appliquant les règles d'opérations sur les limites

- si la première partie a échoué , elle fournit des renseignements qui permettent de choisir une ou plusieurs transformations heuristiques destinées à modifier profondèment la structure de l'expression sur laquelle on réapplique POLLIM.

Bien que le problème soit différent, on peut rapprocher la démarche générale de DALI de celle de SIN : on a d'abord une heuristique puissante et rapide

(calculs faciles et performants sur les polynomes) qui fait le maximum de travail, puis en cas d'échec, les renseignements concernant la structure de l'expression conduisent à utiliser judicieusement certaines heuristiques.

. Vérification d'identités en utilisant le raisonnement par récurrence
VIVET (46) :

L'auteur se propose de résoudre des problèmes d'algèbre, de trigonométrie, de dérivation ou de suites numériques du type

$$\sum_{i=1}^{n} i = n\,\frac{(n+1)}{2} \quad \text{ou} \quad \sum_{i=1}^{n} \cos ix = \frac{\sin n\,\frac{x}{2}}{\sin \frac{x}{2}}\,\cos (n+1)\,\frac{x}{2}$$

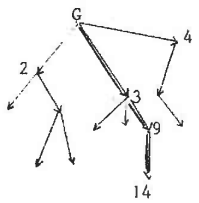
La méthode de résolution choisie est simple :
. vérifier l'égalité au rang 1
. en posant qu'elle est vraie au rang n, prouver qu'elle l'est encore au rang n + 1.

Les techniques de l'intelligence articielle interviennent au niveau de l'élaboration de cette preuve. Par sa nature, ce problème se rattache donc aux démonstrations d'identités déjà envisagées avec toutefois une règle privilégiée à appliquer obligatoirement : l'hypothèse de récurrence.

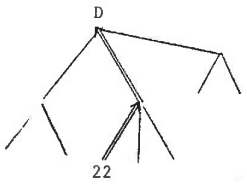
L'originalité de la méthode réside dans l'intervention du membre droit de l'égalité à vérifier au cours de la fabrication de la preuve.

Soit à vérifier G = D

Le programme établit à la fois une arborescence à partir de G en appliquant certaines règles et une arborescence de conjectures à partir de D. Si toutes les règles de réécriture sont symétriques comme c'est le cas pour les calculs algébriques courants, il suffit de trouver 2 feuilles communes à ces 2 arborescences.



Arborescence des dérivés de G



Arborescence des conjectures possibles pour atteindre D

Si les expressions des noeuds 14 et 22 sont identiques, le chemin en trait double donne les étapes de la démonstration.

Ce programme contient également un système de calcul algébrique performant sur lequel nous reviendrons.

CONCLUSION :

En conclusion, l'étude de ces programmes qui systématiquement ou accessoirement travaillent dans un environnement trigonométrique met en évidence deux types d'approche de la résolution heuristique de problèmes :

. l'une connaît la "forme" de la solution et cherche à rapprocher la situation initiale de la situation finale avec des heuristiques plus ou moins générales selon le but recherché ;

. l'autre connaît seulement une définition de la solution, utilise quelques heuristiques performantes quasi algorithmiques pour résoudre un maximum de problèmes, puis possède une liste d'heuristiques plus spécifiques à appliquer avec précaution.

Le premier type d'approche est celui de G.P.S. et DATAL pour les programmes à vocation générale, de VIVET pour un travail plus spécifique ; le deuxième est celui de SAINT, SIN et DALI.

Il est intéressant de noter qu'à ces deux types d'approches différents correspondent des formulations différentes de problèmes : pour les premiers

la "forme" de la solution est parfaitement connue, il s'agit seulement de mettre en évidence un chemin allant du point de départ au point d'arrivée en utilisant les caractéristiques des deux. Pour les seconds au contraire, la solution peut revêtir des formes très diverses et la stratégie est différente.

Un autre problème appartient à cette deuxième catégorie, c'est la résolution d'équations trigonométriques. Aucun article à notre connaissance ne lui a été consacré, c'est pourquoi nous avons choisi de l'étudier.

3ème partie

P.R.E.T.

UN PROGRAMME DE RESOLUTION FORMELLE D'EQUATIONS TRIGONOMETRIQUES

I - NATURE DU PROBLEME - APPROCHES POSSIBLES

1) - Les équations trigonométriques :

Dans de nombreux domaines, géométrie, mécanique, astronomie, physique, etc..., les applications conduisent à des équations où l'inconnue, qui est en général un angle, figure par l'intermédiaire d'une ou plusieurs de ses lignes trigonométriques. De telles équations sont appelées équations trigonométriques, nous nous limiterons dans cette étude aux cas où l'inconnue n'apparaît pas autrement que sous un symbole trigonométrique.

Ainsi par exemple les équations :

$$\cos x = \cos 2x$$

$$\sin x = 2 \sin (x\sqrt{2})$$

$$\cos^2 \frac{x}{2} + \sin x = 1 + \mathrm{tg}^3 x$$

$$(2 - \sin^3 \frac{x}{3}) \quad \sqrt{1 + \cos^2 x} = k$$

seront des "équations trigonométriques", alors que les équations :

$$x + \sin x = \frac{4}{3}$$

$$\sin \pi \sqrt{x} = 2\sqrt{x}$$

$$\log x = \sin x$$

n'en seront pas.

Le problème de la résolution générale des équations trigonométriques n'est pas résolu, il ne l'est que dans des cas très restreints et très simples. Mais on sait résoudre certaines équations souvent rencontrées, et on dispose pour cela d'un certain nombre de procédés.

2) - Les méthodes employées :

Les méthodes employées consistent surtout à ramener la résolution d'une équation trigonométrique à celle d'une équation algébrique ordinaire. On cherche à prendre une inconnue auxiliaire, de manière à faire disparaître les symboles trigonométriques et à se ramener finalement à l'étude de l'une des équations très simples :

$\cos x = a$

$\sin x = b$

$\text{tg } x = c$

$\text{cotg } x = d$

Nous supposerons la résolution de ces équations parfaitement connue du lecteur. On sait également qu'une équation linéaire en sin x et cos x se ramène aisèment aux cas précédents, nous considérons donc comme immédiate la solution des équations du type :

$$A \cos x + B \sin x = C$$

Lorsque l'équation proposée ne se ramène pas immédiatement à l'un des types précédents, on ne peut pas donner de méthode générale.

3) - Approches algorithmiques possibles :

On pourrait néanmoins penser à deux transformations qui paraissent assez générales :

. l'expression complexe des lignes trigonométriques.
Les formules de transformation sont les suivantes :

$$\sin x \longrightarrow \frac{e^{ix} - e^{-ix}}{2i}$$

$$\cos x \longrightarrow \frac{e^{ix} + e^{-ix}}{2}$$

Si on applique systématiquement auparavant les deux règles :

$$\text{cotg } x \longrightarrow \frac{1}{\text{tg } x}$$

$$\text{tg } x \longrightarrow \frac{\sin x}{\cos x}$$

Ce passage aux expressions complexes de sinus et cosinus n'a jamais fourni la méthode simple pour résoudre beaucoup d'équations trigonométriques et il ne semble pas qu'on ait intérêt à l'employer ici.

. si l'équation se présente sous la forme

$P(\cos x, \sin x, \cos a\,x, \cos b\,x, \ldots, \sin a'x, \sin b'x \ldots) = 0$

P étant un polynome, et si les constantes a, b,... , a', b',... sont rationnelles, on peut les réduire au même dénominateur N ; si on

pose alors $\text{tg } \frac{x}{2N} = t$, toutes les expressions précédentes sont rationnelles en t, l'équation prend la forme :

$\mathcal{P}(t) = 0$ et l'on est ramené à un problème d'algèbre.

Cette solution est très puissante au niveau théorique, elle donne un algorithme général pour toutes les équations répondant à la définition ci-dessus. Cependant, le polynome en t obtenu est en général de degré élevé, et l'utilisateur risque bien d'être ramené à une équation qu'il ne sait pas davantage résoudre que la première.

Donnons deux exemples significatifs :

* $\sin 5x + \sin x + 2 \sin^2 x - 1 = 0$

nous donnerait un polynome du 10° degré en $t = \text{tg } \frac{x}{2}$

alors qu'on obtient des solutions simples en transformant d'abord la somme et en linéarisant le carré :

$$2 \sin 3x \cos 2x - \cos 2x = 0$$

soit $\cos 2x (2 \sin 3x - 1) = 0$

dont les solutions sont données par les 2 équations :

$$\begin{cases} \cos 2x = 0 \\ \sin 3x = \frac{1}{2} \end{cases}$$

* $\sin^3 x + \cos^3 x = 1$

donne, en posant $\text{tg } \frac{x}{2} = t$:

$$t^2 (-2t^4 + 8t - 6) = 0$$

alors qu'en remarquant que l'expression est symétrique en sin x et cos x, on peut faire le changement de variable

$x = \frac{\pi}{4} + y$ qui conduit à l'équation

$$2\sqrt{2} \cos^3 y + \frac{3\sqrt{2}}{2} \cos y - 1 = 0 \quad \text{de degré moins élevé.}$$

Il paraît donc intéressant d'étudier et de répertorier les heuristiques qu'utilise le mathématicien pour se ramener à une ou plusieurs équations algébriques simples lorsqu'il a à résoudre une équation trigonométrique.

II - ETUDE DU COMPORTEMENT HUMAIN

Notre but n'est évidemment pas de dresser à travers la résolution de quelques problèmes une liste complète des heuristiques utilisées par le mathématicien. Nous voulons simplement mettre en évidence certaines d'entre elles pour étudier ensuite comment nous pouvons les formaliser et lorsque cela ne sera pas possible ou pas souhaitable voir par quoi les remplacer.

1) - Quelques exemples (présentés sous la forme f (x) = 0) :

Premier exemple :

$$2 \sin x - 4 m \cos^2 x + 3 = 0$$

$\cos^2 x$ s'exprime facilement en fonction de sin x, nous effectuons donc la transformation pour obtenir une équation qui sera algébrique du second degré si nous posons A = sin x, soit :

$$2 \sin x + 4 m \sin^2 x - 4 m + 3 = 0$$

$$4 m A^2 + 2A - 4 m + 3 = 0$$

Deuxième exemple :

$$\cos 2 x - \cos 4 x - \sin x = 0$$

Cos 2 x et cos 4 x s'expriment en fonction de sin x, nous pouvons donc effectuer les transformations qui nous conduiront à une équation algébrique avec pour inconnue auxiliaire A = sin x, soit :

$$\cos 4 x \longrightarrow 2 \cos^2 2x - 1$$

$$\cos 2 x \longrightarrow 1 - 2 \sin^2 x$$

ce qui donne toutes simplifications effectuées :

$$\sin x (- 8 \sin^3 x + 6 \sin x - 1) = 0$$

On est ramené à résoudre deux équations dont l'une est du troisième degré, n'était-il pas possible d'avoir une équation de degré inférieur ?

On peut appliquer aux deux premiers termes du premier membre les formules de transformation de somme en produit, ce qui donne :

$$2 \sin 3 x \sin x - \sin x = 0$$

$$\sin x (2 \sin 3 x - 1) = 0$$

Cette méthode nous donne une solution plus simple puisqu'elle amène à résoudre deux équations du premier degré.

Troisième exemple :

$$\cos x - \cos 2x - \sin 3x = 0$$

Nous savons qu'il n'est pas possible d'exprimer sin 3x en fonction de cos x, nous allons donc chercher à effectuer des mises en facteurs en transformant des sommes de termes ou des termes en produits et en faisant apparaître les mêmes facteurs :

$$\cos x - \cos 2x \longrightarrow 2 \sin \frac{3x}{2} \sin \frac{x}{2}$$

$$\sin 3x \longrightarrow 2 \sin \frac{3x}{2} \cos \frac{3x}{2}$$

d'où $$2 \sin \frac{3x}{2} \sin \frac{x}{2} - 2 \sin \frac{3x}{2} \cos \frac{3x}{2} = 0$$

$$2 \sin \frac{3x}{2} \left(\sin \frac{x}{2} - \cos \frac{3x}{2}\right) = 0$$

Nous obtenons deux équations, la solution de la première est immédiate. Pour résoudre la seconde, nous pouvons encore penser à faire apparaître un produit de facteurs :

$$\sin \frac{x}{2} \longrightarrow \cos \left(\frac{\pi}{2} - \frac{x}{2}\right)$$

d'où $$\cos \left(\frac{\pi}{2} - \frac{x}{2}\right) - \cos \frac{3x}{2} = 0$$

$$- 2 \sin\left(\frac{\pi}{4} + \frac{x}{2}\right) \sin\left(\frac{\pi}{4} - x\right) = 0$$

Nous sommes ainsi ramenés à résoudre 3 équations très simples.

Quatrième exemple :

$$\sin^2 4x - \sin^2 x = 0$$

Transformons la différence de deux carrés :

$$(\sin 4x + \sin x) (\sin 4x - \sin x) = 0$$

Ce qui donne deux équations aux premiers membres desquelles nous appliquons les formules de transformation de somme en produit :

$$2 \sin \frac{5x}{2} \cos \frac{3x}{2} = 0$$

$$2 \sin \frac{3x}{2} \cos \frac{5x}{2} = 0$$

On est ramené à 4 équations simples.

2) - Nature des heuristiques utilisées :

Dans les deux premiers cas, c'est notre connaissance des formules trigonométriques qui nous guide. Il existe pourtant des propriétés mathématiques que vérifient les expressions premiers membres des équations et qui nous indiquent qu'on peut les exprimer en fonction de sin x ou cos x.
Dans les second et troisième cas, nous sommes guidés à la fois par la forme de l'équation et par l'idée de faire apparaître un facteur commun.
Dans le quatrième cas, nous reconnaissons une identité remarquable algébrique et nous l'utilisons.
L'étude de nombreux autres cas montre que les heuristiques que nous utilisons, même si elle nous apparaissent comme essentiellement "visuelles", peuvent être classées en trois types :
Les unes reposent sur des propriétés mathématiques de l'expression f(x).
Dans d'autres cas, nous reconnaissons dans tout ou partie de l'expression f (x) le premier membre d'une transformation et nous percevons que le deuxième membre de cette transformation nous amènera à une situation plus avantageuse.
Enfin, dans des situations plus compliquées nous appliquons localement certaines transformations avec un but assez vague comme "faire apparaître un facteur commun" sans préciser lequel ou plus précis comme "réduire au même argument" qui peut fournir une étape intermédiaire pour réaliser le précédent.

3) - Qu'est-ce qu'une solution simple ? :

Nous avons vu qu'il ne suffisait pas d'avoir moyennant un ou plusieurs changements de variables ramené la résolution de l'équation trigonométrique donnée à la résolution d'une ou plusieurs équations algébriques pour que le problème soit considéré comme achevé. Le but est d'obtenir les équations les plus simples, c'est-à-dire de degré le moins élevé possible, parce que les calculs formels conduisant à l'expression de la solution seront moins nombreux et l'expression même de cette solution sera plus facile.

Voyons maintenant comment déterminer des heuristiques qui permettent à un programme d'atteindre ce but.

III - UTILISATION HEURISTIQUE DE PROPRIETES MATHEMATIQUES ET "FORMELLES"

Certaines propriétés mathématiques d'une expression f (x) peuvent indiquer que des règles données améneront à la solution en transformant l'équation en une autre qui sera plus facile à résoudre. Il ne faudra cependant pas les utiliser sans discernement. Nous allons donc les rappeler puis montrer comment un programme peut s'en servir.

1) - Rappel des résultats concernant une expression trigonométrique f (x) :

Nous adoptons dans toute la suite de l'exposé pour f (x) une définition analogue à celle des équations trigonométriques, f (x) est une expression dans laquelle l'inconnue x ne figure que par l'intermédiaire d'une ou plusieurs de ses lignes trigonométriques. On pourra trouver les preuves des résultats que nous avançons ici dans (1), (2), (3) ou (4).

(a) Si f (x) ne change pas lorsqu'on change x en $x + \pi$ et si f (x) change de signe avec x, alors on peut exprimer rationnellement f (x) en fonction de tg x.

(b) Si f (x) change de signe avec x, mais ne change pas si l'on change x en $\pi - x$, alors on peut l'exprimer rationnellement en fonction de sin x.

(c) Si f (x) ne change pas de signe avec x, mais change si x devient $\pi - x$, alors on peut l'exprimer rationnellement en fonction de cos x.

(d) Si f (x) ne change pas de signe quand x devient - x et quand x devient $x + \pi$, alors on peut l'exprimer rationnellement en fonction de cos 2 x.

(e) Si f (x) est de la forme g (sin x, cos x, tg x, ...), en remplaçant tg x et cotg x en fonction de sin x et cos x, on obtient une expression du type F (sin x, cos x).

(f) Si f (x) est de la forme P (sin x, cos x), où P est un polynome homogène en sin x et cos x, en plus généralement si les degrés de tous ses termes

par rapport à sin x et cos x sont de même parité, on obtient en divisant tous les termes par $\cos^n x$, n étant le degré de P, une expression en tg x.

(g) Si f (x) est de la forme F (sin x, cos x) avec F symétrique en sin x et cos x, les racines de l'équation f (x) = 0 sont deux à deux symétriques par rapport à $\frac{\pi}{4}$. On a donc intérêt à faire le changement de variable $x = \frac{\pi}{4} + y$.

2) - Utilisation de ces propriétés :

Ces propriétés semblent nous fournir des méthodes algorithmiques de résolution dans de nombreux cas. Il faut pourtant s'en méfier comme du passage à l'inconnue $t = tg \frac{x}{2}$. Rien ne prouve en effet que l'équation en sin x ou tg x promise au moment de l'analyse de l'expression soit simple.
Le deuxième exemple résolu au paragraphe précédent montre que l'idée d'exprimer rationnellement l'expression en fonction de sin x ne conduisait pas à la solution la plus simple.

Nous avons donc donné l'ensemble de ces propriétés à notre programme, il en fera largement usage soit pour appliquer directement certaines règles, soit pour choisir parmi plusieurs celles qui lui sembleront favorables. Par exemple, s'il existe dans l'expression à transformer les arguments x et 4x ou 5x, le programme n'essaiera pas d'abord d'obtenir une équation algébrique en sin x car elle serait de degré élevé. Toutefois, cette possibilité ne sera pas définitivement exclue, en effet une équation de degré élevé peut se ramener parfois à des équations très simples, le programme y reviendra par la suite s'il n'a pas trouvé de solution par d'autres moyens.

Il reste à définir l'ordre dans lequel il est souhaitable que ces propriétés soient utilisées. L'étude de nombreux exemples nous a amenés à l'imposer au programme de façon à éviter au maximum les "fausses pistes" ou les essais stériles. Le choix de cet ordre est donc heuristique.

On pourrait sans doute concevoir un programme qui déterminerait statistiquement cet ordre lui-même.

L'ensemble de ces remarques nous ont permis de dresser la liste de règles obligatoires qui suit. Pour l'ordre d'application de ces règles, on se reportera à l'organigramme B à la fin de ce chapitre.

3) - Liste de règles obligatoires - R.O. :

a - Supprimer les radicaux qui portent sur une expression contenant la variable par élévations successives au carré.

b - Si f (x) contient des tangentes et des cotangentes et si toutes les lignes trigonométriques ne portent que sur l'argument x, appliquer la transformation $\text{cotg } x \longrightarrow \frac{1}{tgx}$.

c - Si f (x) contient des tangentes ou des cotangentes (ou inclusif) :

c1 - si toutes les fonctions trigonométriques portent sur le même argument et si f (x) contient des sinus ou des cosinus, appliquer les transformations $tgx \longrightarrow \frac{\sin x}{\cos x}$ et $\text{cotg } x \longrightarrow \frac{\cos x}{\sin x}$ pour obtenir une équation de la forme F (sin x, cos x) = 0.

c2 - si toutes les fonctions trigonométriques ne portent que sur deux arguments (x et 2 x) et si f (x) ne contient pas sin x ou cos x à une puissance impaire, appliquer les transformations qui permettent d'aboutir à une équation de la forme F (tg x) = 0.

d - Si f (x) est de la forme P (sin x, cos x) :

d1 - si sin x (resp. cos x) ne figure dans P qu'avec des puissances paires, appliquer la transformation $\sin^2 x \longrightarrow 1 - \cos^2 x$ (resp. $\cos^2 x \longrightarrow 1 - \sin^2 x$).

d2 - si tous les termes de P sont de même parité n par rapport à sin x et cos x, diviser P par $\cos^n x$ et appliquer les transformations $\frac{\sin x}{\cos x} \rightarrow tg\ x$, $\frac{\cos x}{\sin x} \longrightarrow \frac{1}{tg\ x}$, $\cos^2 x \rightarrow \frac{1}{1+tg^2 x}$ pour obtenir une équation de la forme F (tg x) = 0.

d3 - si P est symétrique en sin x et cos x, effectuer le changement de variable $x \longrightarrow x + \frac{\pi}{4}$.

e - Si f (x) ne contient que x et 2 x comme arguments de fonctions trigonométriques, essayer d'obtenir une équation de la forme F (cos 2 x) = 0 ou F (cos x) = 0 ou F (sin x) = 0 ou F (tg x) = 0 par application des transformations qui permettent d'aboutir au but correspondant.

f - Si f (x) a la forme du premier membre de l'une des transformations algébriques suivantes, appliquer cette transformation :

$u^2 - v^2 \longrightarrow (u + v)(u - v)$

$u^3 - v^3 \longrightarrow (u^2 + uv + v^2)(u - v)$

g - Si f (x) est de la forme $\sin u \pm \cos v$ ou $\mathrm{tgu} \pm \mathrm{cotg}\ v$, appliquer les transformations :

$\sin u \longrightarrow \cos (\frac{\Pi}{2} - u)$

ou

$\mathrm{cotg}\ u \longrightarrow \mathrm{tg}\ (\frac{\Pi}{2} - u)$

h - Si f (x) est de la forme trig (u) $\pm$ trig (v) où trig représente l'une des quatre lignes trigonométriques sin, cos, tg, cotg, transformer f (x) en un produit de facteurs par application des transformations correspondantes.

i - Si aucune des règles numérotées de a à h n'est plus applicable et s'il reste des dénominateurs, réduire f (x) au même dénominateur et conserver le numérateur de la fraction obtenue.

La formulation de ces règles appelle deux remarques.

D'une part, certaines règles ont été regroupées autour d'une même lettre (c, d, e, etc...) pour des commodités d'exposé et de programmation. En effet l'expression f (x) doit vérifier un critère commun pour qu'on puisse essayer de les appliquer. Elles constituent cependant des règles indépendantes.

D'autre part, certaines règles engendrent un but à atteindre, par exemple : "obtenir une équation de la forme F (tg x) = 0". Le programme ne considérera ces règles comme "applicables" que si elle permettent effectivement d'atteindre le but fixé. Les "buts" possibles sont au nombre de quatre :

obtenir une équation de la forme F (tg x) = 0
" " " F (cos 2 x) = 0
" " " F (cos x) = 0
" " " F (sin x) = 0

Le programme dispose d'une table lui indiquant quelles transformations peuvent lui permettre d'atteindre ces buts (cf. annexe D à la fin de cette troisième partie). Cette table est volontairement très réduite, nous verrons que les règles obligatoires s'appliquent souvent et doivent être rapides.

Enfin, nous n'avons pas voulu surcharger l'énoncé de ces règles par trop de formules de trigonométrie, l'ensemble des transformations utilisées par le programme tant dans les règles obligatoires que dans les règles non obligatoires est donné à l'annexe E de cette troisième partie.

Nous énonçons maintenant la seconde liste des règles retenues puis nous expliquerons comment le programme P.R.E.T. les utilise.

4) - Liste de règles non obligatoires - R.N.O. :

a - Appliquer les formules de transformation de somme en produit (Groupe I).

b - Si f (x) contient des éléments de la forme sin m x, avec m $\geq$ 4 et m pair, appliquer la transformation $\sin mu \longrightarrow 2 \sin \frac{m}{2} u \cos \frac{m}{2} u$

Sinon, appliquer la transformation $\sin 2u \longrightarrow 2 \sin u \cos u$.

c - Appliquer les formules de transformation de produit en somme (Groupe II).

d - Développer les lignes trigonométriques d'une somme (Groupe III).

e - Appliquer la transformation $\mathrm{cotg}\ u \longrightarrow \frac{1}{\mathrm{tgu}}$.

f - Si sin x, cos x, tg x ne figurent pas sans exposant dans f (x), linéariser (Groupes VII et XIII).

g - Si f (x) comporte plus d'un argument, ramener tous les arguments à x (Groupes VI-1 et VI-2).

h - Appliquer localement les transformations algébriques prévues pour les R.O.

i - Ramener les expressions en tangente et cotangente à leur équivalent en sinus et cosinus (Groupe IX).

j - Si f (x) est de la forme F (sin x, cos x), effectuer le changement de variable $x = x + \frac{\pi}{4}$ si les règles a...i ne sont pas pas applicables.

k - Si f (x) est de la forme F (sin x, cos x) effectuer le changement de variable x = 2 x et appliquer les transformations du Groupe VIII si les règles a...j ne sont pas applicables.

Les numéros indiqués pour les groupes de formules renvoient à l'annexe 4. Toutes les règles non obligatoires applicables à une étape de la résolution sont appliquées, elles génèrent donc de nouvelles formes de l'équation à résoudre. L'objet du paragraphe qui suit est d'indiquer comment le programme organise cet ensemble d'équations et utilise les R.O. et R.N.O. que nous venons de donner.

IV - LA DEMARCHE GENERALE DU PROGRAMME

P.R.E.T. opère de façon récursive, il applique une transformation à f (x), génère une ou plusieurs nouvelles équations auxquelles il appliquera les mêmes méthodes qu'au stade précédent. A chaque stade de la résolution, les opérations à effectuer appartiennent à l'un des trois types suivants :

. f (x) peut prendre la forme d'une équation algébrique simple après un changement d'inconnue approprié ; on considère alors que P.R.E.T. a résolu cette équation, il suffira d'utiliser un algorithme de résolution d'équation algébrique pour obtenir les solutions effectives. Il lui reste éventuellement à résoudre les équations en attente.

. une règle obligatoire est applicable à f (x), cette règle est toujours appliquée. P.R.E.T. génère une nouvelle équation qu'il se propose de résoudre.

. dans les autres cas, on applique une ou plusieurs règles non obligatoires, ce qui génère une ou plusieurs nouvelles équations.

Nous donnons à la fin de cette partie (annexe A) le schéma général de fonctionnement du programme. Nous allons le décrire plus précisément ici.

1) - Organisation des équations générées :

Nous venons de voir qu'à partir de l'équation donnée, le programme génère de nouvelles équations par application de transformations. Si nous considérons l'une de ces équations intermédiaires, trois cas peuvent se présenter pour la transformation de cette équation :

a - Une règle obligatoire est applicable ou bien une seule règle non obligatoire est applicable. L'application de la règle génère une équation qu'il faudra résoudre à la place de celle dont elle est issue.

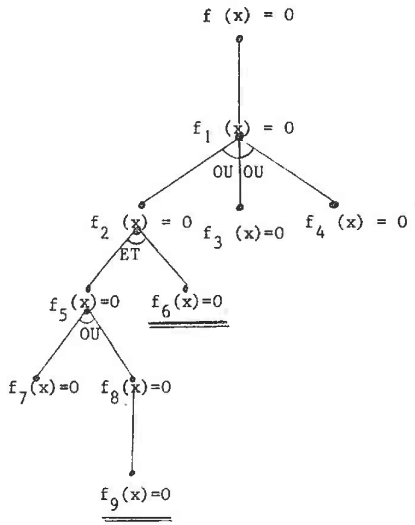
b - Le premier nombre f (x) de l'équation a la forme d'un produit de p facteurs contenant l'inconnue x. Pour résoudre f (x) = 0, il faudra résoudre les p équations obtenues en égalant les p facteurs à 0. Dans ce cas la résolution d'une équation est remplacée par la résolution de plusieurs autres.

c - Plusieurs règles non obligatoires sont applicables à f (x). L'application de chacune d'elle génère une nouvelle équation ; pour résoudre l'équation f (x) = 0 il faudra résoudre l'une des équations générées par application de règles non obligatoires à f (x).

Il est commode de représenter l'ensemble des équations ainsi générées par une arborescence ET/OU. A chaque noeud de l'arborescence seront attachées une équation et une marque correspondant à l'un des trois cas précédents.

Une marque ET signifie que pour avoir la solution de l'équation représentée par ce noeud, il faut avoir résolu toutes celles représentées par les noeuds "fils". Une marque OU signifie qu'il faut avoir résolu l'une des équations représentées par les noeuds fils. L'absence de marque signifie qu'une seule branche est issue du noeud considéré.

Donnons un exemple simple :



L'équation à résoudre est f (x) = 0. Elle est transformée en f_1 (x) = 0 qui donne naissance à trois équations possibles.
L'étude de la première f_2 (x) = 0 conduit à la décomposer en deux équations f_5 (x) = 0 et f_6 (x) = 0. Pour résoudre f_5 (x) = 0 on peut étudier f_7 (x) = 0 ou f_8 (x) = 0, cette dernière donne après transformation f_9 (x) = 0 dont on a une solution immédiate. Il reste donc à résoudre f_6 (x) = 0 dont on a aussi une solution immédiate. On a ainsi résolu f (x) = 0.

Nous donnons à la fin du chapitre les arborescences générées pour chacun des exemples traités. Précisons maintenant les liens entre application des règles R.O. et R.N.O. et création de noeuds dans l'arborescence.

2) - Application des règles et génération des noeuds :

Examinons d'abord le cas où P.R.E.T. a résolu une équation.

Solution immédiate :

P.R.E.T. considère qu'il a résolu une équation lorsque celle-ci ne comporte plus qu'une seule ligne trigonométrique portant sur un seul argument et qu'en prenant cette ligne comme inconnue auxiliaire, il arrive à une équation des 1, 2, 3ème degré ou bicarrée. Il considère également comme immédiate la solution des équations du type a cos u + b sin u + c = 0.

Dans les cas où la solution n'est pas immédiate, P.R.E.T. peut appliquer plusieurs types de transformations. Nous avons déjà établi une distinction entre règles obligatoires et règles non obligatoires ; un troisième type de transformation est très important pour simplifier les résolutions d'équations trigonométriques, il s'agit de la mise sous forme d'un produit de facteurs de l'expression f (x). Etudions donc ces trois types de transformations :

Mise en facteur :

Lorsqu'une nouvelle équation est générée, P.R.E.T. essaie de la mettre sous forme d'un produit de facteurs. Pour cela, il dispose d'un sous-programme MIFAC qui opère de la façon suivante :

f (x) est une somme de termes : $f(x) \sum_{i=1}^{n} Ti$.

chaque terme Ti est un produit de facteurs ou un quotient de facteurs :

$$Ti = \prod_{j=1}^{p} Tip$$

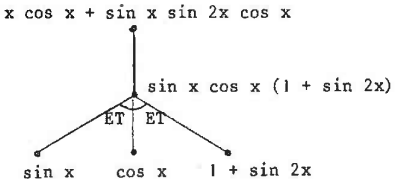
MIFAC examine un à un les facteurs de T_1 et cherche à les retrouver dans les autres termes. Lorsqu'un facteur apparaît dans tous les termes, il est mis en facteur. Soit $T_1 p_0$ ce facteur, f (x) s'écrit :

$$f(x) = T_1 p_0 * \sum_{i=1}^{n} T'i$$

où les T'i sont obtenus à partir des Ti en enlevant le facteur $T_1 p_0$.
MIFAC recommence la même opération sur la somme $\sum_{i=1}^{n} T'i$ pour mettre en évidence un éventuel second facteur commun ; et ainsi de suite jusqu'à épuisement.
Lorsque MIFAC ne trouve plus de nouveau facteur commun, un nouveau noeud est généré dans l'arborescence représentant la nouvelle forme de f (x).

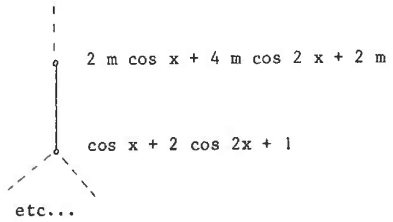
A partir de ce noeud N, chacun des facteurs contenant l'inconnue donne naissance à un nouveau noeud. Pour résoudre l'équation représentée par N, il faut évidemment résoudre toutes celles qui en sont issues, N est un noeud ET dans l'arborescence de résolution.

Exemple : Soit à résoudre $\sin x \cos x + \sin x \sin 2x \cos x = 0$
L'application de MIFAC engendre l'arborescence suivante :



Si MIFAC a permis de mettre en évidence un ou plusieurs facteurs constants, on engendre un noeud représentant l'équation obtenue en supprimant ces facteurs constants.

Exemple : Soit à résoudre $2 m \cos x + 4 m \cos 2x + 2 m = 0$
L'application de MIFAC engendre l'arborescence suivante :



Pour que MIFAC fonctionne correctement, il faut lui donner une expression dans laquelle les "facteurs" à chercher sont en évidence. Nous reviendrons dans la quatrième partie sur les transformations à appliquer à f(x) pour la présenter à MIFAC.

Après application de MIFAC et génération d'un noeud ET, le programme continue avec l'équation feuille située la plus à gauche.

Règles obligatoires :

Lorsqu'aucune mise en facteur n'est possible sur une expression, P.R.E.T. essaie d'appliquer une "règle obligatoire". Les essais d'application se font dans l'ordre indiqué à l'organigramme B (cf. annexe B en fin de cette troisième partie) en commençant toujours par le début.

Certaines règles demandent d'appliquer plusieurs transformations, ces transformations sont toutes appliquées en tout point de l'expression jusqu'à ce qu'aucune d'entre elles ne soit plus applicable.

Exemple : soit à résoudre $\sqrt{\sin x} + \sqrt{\cos x} - 1 = 0$.
La première règle obligatoire applicable est la règle a. Au cours de l'application deux élévations successives au carré seront effectuées afin d'éliminer les deux radicaux.

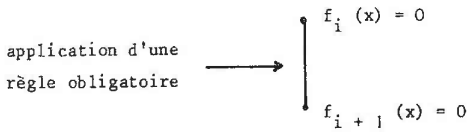
Certaines règles sont regroupées sous une même lettre parce que leur application demande au préalable la vérification d'un critère commun, par exemple "f(x) contient des tangentes et des cotengentes". Une fois ce critère vérifié, les règles indiquées par des tirets sont indépendantes et le programme tente d'en appliquer une seule dans l'ordre où elles se présentent.

Exemple : soit à résoudre $2 \sin^2 x + \sin^2 2x - 2 = 0$
La première règle obligatoire applicable est la règle e. Le programme va chercher à obtenir une équation de la forme F (cos 2x) = 0 par application des transformations correspondantes (cf. annexe D en fin de cette troisième partie). S'il échoue, il essaiera d'obtenir une équation de la forme F (cos x) = 0, etc...

Ici, on obtient une équation F (cos 2 x) = 0 : $\cos^2 2x - \cos 2x = 0$
Comme nous l'avons indiqué, le programme essaie ensuite de faire des mises en facteur sur cette nouvelle équation.

Chaque fois qu'une règle obligatoire est appliquée avec succès, le programme génère un noeud "fils" du noeud sur lequel on a appliqué cette règle.

Rappelons que notre liste de règles obligatoires est heuristique, cependant nous avons rangé dans cette liste celles des transformations qui paraissent devoir être systématiquement appliquées. En conséquence si $f_i(x) = 0$ est l'équation à laquelle on applique une règle R.O. et si $f_{i+1}(x) = 0$ est l'équation transformée, le programme ne travaillera plus avec $f_i(x)$ mais avec $f_{i+1}(x)$. Cela revient à n'admettre qu'une seule branche issue du noeud $f_i(x)$ dans l'arborescence de résolution selon le schéma suivant :



Règles non obligatoires :

Lorsqu'aucune règle obligatoire n'est plus applicable à aucune expression le programme sélectionne une équation à laquelle on essaie d'appliquer les unes après les autres les règles non obligatoires.

Comme pour les règles obligatoires, lorsqu'une règle non obligatoire comporte l'application de plusieurs transformations, ces transformations sont appliquées en tout point de l'expression tant qu'elles sont applicables.

Exemple : tg x - tg 3x + cotg x - cotg 3x - 4 = 0.
Avec la règle non obligatoire a, on transforme à la fois
tg x - tg 3x et cotg x - cotg 3x.

Cependant l'application de certaines règles soulève des ambiguités. Soit l'équation tg x + tg 2x + tg 3x = 0.
La règle non obligatoire a est applicable, elle peut permettre de transformer soit tg x + tg 2x, soit tg x + tg 3x, soit tg 2x + tg 3x.
Ce problème se pose chaque fois que les transformations applicables ont comme membre de gauche une somme de deux termes et que l'équation à résoudre se présente comme une somme de plusieurs termes.

Dans ce cas le programme considère tous les cas possibles, ce qui équivaudra à plusieurs applications différentes de la même règle.

L'application d'une règle non obligatoire donne donc naissance à un ou plusieurs noeuds dans l'arborescence de résolution.

Exemple : pour un noeud portant tg x + tg 2x + tg 3 x = 0
l'application de la R.N.O. a donne naissance à trois noeuds correspondant aux trois possibilités envisagées ci-dessus :

(un exemple analogue est entièrement analysé au paragraphe suivant).

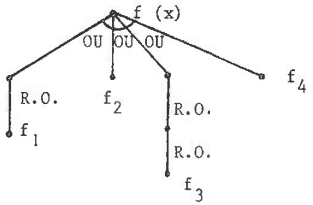
Lorsque l'application à un noeud N d'une ou plusieurs règles non obligatoires amène à créer plusieurs branches issues de N, le noeud N est marqué OU.

Dès qu'une R.N.O. a été appliquée ou essaie d'appliquer toutes les R.O. possibles à l'équation obtenue, ceci dans le but de ne pas essayer d'autres règles si une solution immédiate ou une simplification importante est possible.

. Simplifications - Normalisation :

Après chaque application d'une règle, l'expression obtenue subit certaines transformations pour la simplifier et la mettre sous une forme standard. Ces transformations systématiques ne sont pas comptabilisées au niveau des noeuds de l'arborescence, c'est évidemment le résultat simplifié qui est retenu dans chacun des cas. Nous expliquons les simplifications dans la quatrième partie.

3) - Choix d'un nouveau noeud - Critère de complexité :



Lorsqu'on ne peut plus appliquer aucune règle non obligatoire à une équation f (x) = 0 représentée par un noeud OU dans l'arborescence, on est conduit à choisir dans l'arborescence celle des équations qui présente les meilleures chances de succès pour la suite des calculs.

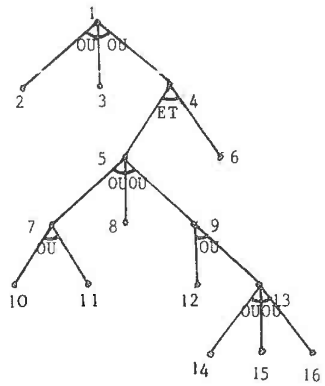
Pour permettre ce choix, le programme évalue la complexité de toute expression à laquelle il ne peut plus appliquer de règle obligatoire. Il nous a semblé nécessaire de prendre en compte le nombre de termes de l'expression et le nombre de facteurs de ces termes, le nombre d'arguments différents et les puissances auxquelles sont élevées les lignes trigonométriques.
Nous avons commencé à travailler avec une évaluation très simple de la complexité :

$$C = N \text{ facteurs} + N \text{ arguments} + \Sigma \text{ puissances}$$

Ce qui nous donne par exemple :
pour $\sin 3x \sin x + \sin^2 2x + \cos x$, $C = 4 + 3 + 2 = 9$
pour $2 \cos 3x \sin^2 x + \cos x \sin^2 2x - tgx$, $C = 6 + 3 + 4 = 13$

Il semble qu'on puisse avoir une expression plus fine de la complexité qui tienne compte en particulier non seulement des différences d'arguments mais aussi de la complexité de ces arguments. Cela n'est pas très grave, si le programme choisit d'étudier une équation trop compliquée, il échouera très vite.

Le programme choisit donc dans l'arborescence parmi les feuilles issues d'une même branche du dernier noeud ET (ou s'il n'y a pas de ET du noeud initial) celle qui présente la complexité la plus faible. En cas d'égalité, l'équation la plus à gauche est sélectionnée.
On peut donc avoir des configurations comme la suivante :



Si après étude du noeud 13, aucune R.N.O. n'est plus applicable, le choix se fait non seulement parmi les descendants de 13, soit 14, 15, 16 mais aussi parmi les autres feuilles issues de la première branche de 4 (dernier noeud ET) soit 10, 11, 8, 12.

Lorsqu'un noeud porte une équation dont la solution est immédiate, le programme cherche si dans l'arborescence il existe des branches de noeuds ET non traités. Cette recherche se fait en partant du noeud ET le plus récent, pour chaque noeud ET les branches sont traitées de gauche à droite. S'il n'y a pas de noeud ET ou s'ils ont tous été traités, la question initiale est résolue.

4) - <u>Suppression d'un noeud</u> :

Lorsqu'aucune R.N.O. n'est applicable à un noeud, il faut évidemment le supprimer de l'arborescence et supprimer également en remontant les autres noeuds devenus inutiles.

En théorie, tous les autres noeuds devraient être conservés.
En pratique, l'expérience montre qu'au delà de certaines limites concernant la longueur ou la complexité il est inutile de vouloir retransformer une expression. Ces limites peuvent bien entendu varier selon la nature des équations qu'on traite. Nous avons retenu comme seul critère le nombre de symboles d'une expression, au delà d'un certain seuil S, le noeud correspondant est abandonné.

Par ailleurs, une arborescence trop grande pourrait encombrer la mémoire ; il faut donc lorsque toute la zone mémoire allouée à l'arborescence est occupée pouvoir supprimer les feuilles présentant les complexités les plus élevées et éventuellement les noeuds correspondants en amont. Ceci ne nous est jamais arrivé.

Enfin des transformations différentes peuvent conduire à des feuilles semblables. Ceci est rare, mais il faut vérifier qu'une équation n'a pas déjà été générée avant de la placer dans l'arborescence. Une première vérification rapide consiste à comparer les complexités, la vérification effective n'est faite que si les complexités sont égales.

5) - Conclusion :

La méthode que nous venons de décrire s'apparente à celle de SLAGLE dans SAINT. Cependant, elle en diffère au niveau des noeuds ET puisqu'une mise en facteurs est considérée par P.R.E.T. comme règle privilégiée et impose de travailler sur les différents facteurs. L'ensemble de l'arborescence doit cependant être conservé, en effet si l'une des équations issues d'un noeud ET s'avère impossible à résoudre, il faudra supprimer ce noeud ET et étudier d'autres équations.

Par ailleurs, les transformations obligatoires se sont révélées efficaces puisqu'elles permettent de résoudre à elles seules plus de la moitié des exercices proposés.

V - EXEMPLES COMMENTES

Nous donnons en annexe à la fin de l'ouvrage des listings correspondant à un certain nombre de solutions élaborées par le programme, ainsi que la liste de toutes les équations résolues. Ici nous allons montrer sur quelques exemples comment s'applique la méthode que nous venons de décrire.

Exemple 1 :

$$4\cos^3 x \sin 3x + 4\sin^3 x \cos 3x - 3 = 0 \qquad (I)$$

Aucune transformation obligatoire n'est applicable à l'expression.
Le programme applique les unes après les autres celles des règles non obligatoires qui sont applicables. Après chaque application d'une R.N.O., il y a essai de mise en facteurs et application des règles obligatoires au noeud généré.

RNO f : (linéariser) $\cos x \sin 3x + \sin x \cos 3x - 1 = 0$ (II)

Aucune règle obligatoire n'est applicable. On calcule la complexité de cette équation : C = 7.

RNO g : (ramener tous les arguments à x)

$$12\cos^3 x \sin x - 12\sin^3 x \cos x - 3 = 0 \qquad (III)$$

Mise en facteur de la constante 3 :

$$4\cos^3 x \sin x - 4\sin^3 x \cos x - 1 = 0 \qquad (IV)$$

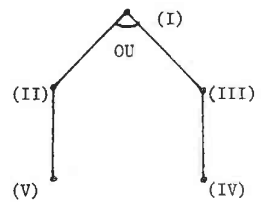
Aucune règle obligatoire n'est applicable, le calcul de la complexité donne C = 13.

Aucune autre règle non obligatoire n'est applicable à l'équation (I).
Le programme cherche à retransformer celle des équations générées dont la complexité est la plus petite, c'est ici l'équation II.
On applique les unes après les autres à (II) celles des règles non obligatoires qui lui sont applicables :

RNO e : $\sin 4x - 1 = 0$ (V)

Cette équation ne comporte qu'une ligne trigonométrique portant sur un seul argument, sa solution est immédiate.

L'arborescence de résolution développée est la suivante :



Exemple 2 :

$$\operatorname{tg} x + \operatorname{tg} 2x + \operatorname{tg} 3x + \operatorname{tg} 4x = 0 \qquad (I)$$

Aucune règle obligatoire n'est applicable. Parmi les transformations non obligatoires on peut appliquer :

RNO a : 1ère application

$$\frac{\sin 7x}{\cos 3x \cos 4x} + \frac{\sin 3x}{\cos x \cos 2x} = 0 \qquad (II)$$

Une règle obligatoire est applicable :

RO i : (réduction au même dénominateur)

$$\sin 7x \cos x \cos 2x + \sin 3x \cos 3x \cos 4x = 0 \qquad \text{(III)}$$

Aucune autre règle obligatoire n'est applicable, le calcul de la complexité donne C = 11.

On continue à appliquer des règles non obligatoires à (I).

RNO a : 2ème application

$$\frac{\sin 6x}{\cos 2x \cos 4x} + \frac{\sin 4x}{\cos x \cos 3x} = 0 \qquad \text{(IV)}$$

RO i : $\sin 6x \cos x \cos 3x + \sin 4x \cos 2x \cos 4x = 0$ (V)

Calcul de la complexité : C = 11.

On continue à appliquer des règles non obligatoires à (I).

RNO a : 3ème application

$$\frac{\sin 5x}{\cos x \cos 4x} + \frac{\sin 5x}{\cos 2x \cos 3x} = 0 \qquad \text{(VI)}$$

Mise en facteur :

$$\sin 5x \left[\frac{1}{\cos x \cos 4x} + \frac{1}{\cos 2x \cos 3x}\right] = 0 \text{(VII)}$$

Une mise en facteur est effectuée, le programme continue donc obligatoirement avec (VII).
Le noeud (VII) de l'arborescence de résolution est marqué ET, deux branches sont issues de ce noeud. La première porte l'équation

$$\sin 5x = 0 \qquad \text{(VIII)}$$

La solution est immédiate.

Le programme revient au noeud ET dont (VIII) est issu, soit (VII).
La deuxième branche issue de ce noeud porte l'équation

$$\frac{1}{\cos x \cos 4x} + \frac{1}{\cos 2x \cos 3x} = 0 \text{ (IX)}$$

RO i : $\cos 2x \cos 3x + \cos x \cos 4x = 0$ (X)

Aucune autre transformation obligatoire n'est applicable. Le programme continue avec (X) puisque c'est le seul noeud possible depuis le dernier noeud ET.

Parmi les règles non obligatoires, on peut appliquer :

RNO c : $\cos 5x + \frac{1}{2} \cos 3x + \frac{1}{2} \cos x = 0$ (XI)

Aucune règle obligatoire n'est applicable.

RNO g : $16 \cos^5 x - 18 \cos^3 x + 4 \cos x = 0$ (XII)

Mise en facteur : $2 \cos x \left[8 \cos^4 x - 9 \cos^2 x + 2\right] = 0$ (XIII)

L'expression se présente comme un produit de facteurs, le noeud (XIII) est donc marqué ET.
La première équation issue de ce noeud est :

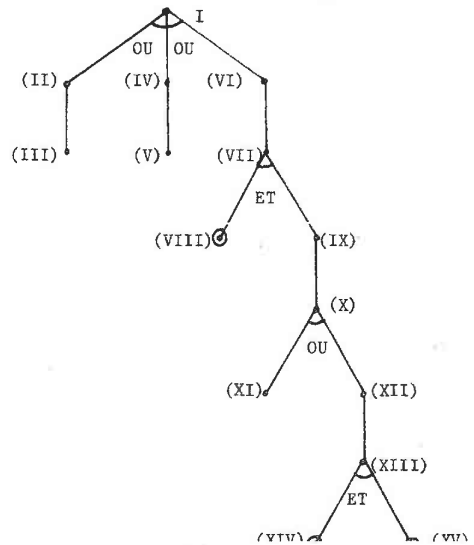
$$\cos x = 0 \qquad \text{(XIV)}$$

Sa solution est immédiate.
On revient au dernier noeud ET, soit (XIII)
Une autre équation issue de ce noeud est :

$$8 \cos^4 x - 9 \cos^2 x + 2 = 0 \qquad \text{(XIV)}$$

Sa solution est immédiate.
Il ne reste aucune branche non explorée à partir des différents noeuds ET.
L'équation (I) est donc résolue.

L'arborescence de résolution développée est la suivante :



Exemple 3 :

$$\sin (n + 1)x + \sin (n - 1)x - \sin 2x = 0 \qquad (I)$$

Aucune règle obligatoire n'est applicable. Parmi les règles non obligatoires

RNO a : $2 \sin n x \cos x - \sin 2x = 0 \qquad (II)$

Aucune règle obligatoire n'est applicable, le calcul de la compléxité donne C = 7.

La règle a est encore applicable de deux façons différentes à (I) :

RNO a : $2 \sin \frac{(n - 1)x}{2} \cos \frac{(n + 3)x}{2} + \sin (n - 1)x = 0 \qquad (III)$

Aucune règle obligatoire n'est applicable, le calcul de la complexité donne C = 7.

RNO a : $2 \sin \frac{(n - 3)x}{2} \cos \frac{(n + 1)x}{2} + \sin (n + 1)x = 0 \qquad (IV)$

Aucune règle obligatoire n'est applicable, le calcul de la complexité donne C = 7.

RNO b : $\sin (n + 1)x + \sin (n - 1)x - 2 \sin x \cos x = 0 \qquad (V)$

Aucune règle obligatoire n'est applicable, le calcul de la complexité donne C = 8.

La RNO d ne s'applique pas ici car l'argument du sin est donné comme un produit et non comme une somme.
Parmi les équations de complexité 7, le programme choisit la première générée, c'est-à-dire (II). Parmi les règles non obligatoires :

RNO b : $2 \sin n x \cos x - 2 \sin x \cos x = 0 \qquad (VI)$

Mise en facteur : $2 \cos x \left[\sin n x - \sin x \right] = 0 \qquad (VII)$

Cette équation est sous le forme d'un produit de facteurs, le noeud (VII) est marqué ET, la première feuille issue de (VII) porte l'équation :

$$\cos x = 0 \qquad (VIII)$$

Sa solution est immédiate. Le programme continue avec la deuxième feuille issue du dernier noeud ET :

$$\sin n x - \sin x = 0 \qquad (IX)$$

Une règle obligatoire est applicable :

RO h : $2 \sin \frac{(n - 1)}{2} x \cos \frac{(n + 1)}{2} x = 0 \qquad (X)$

Cette équation est sous la forme d'un produit de facteurs, le noeud (X) est marqué ET ; la première feuille porte l'équation :

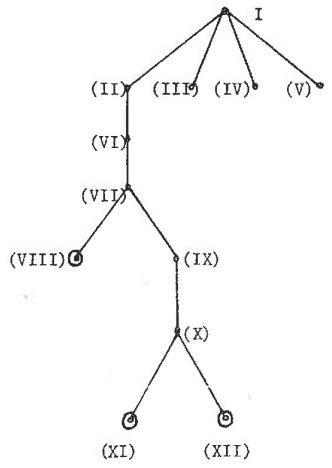
$$\sin \frac{n - 1}{2} x = 0 \qquad (XI)$$

Sa solution est immédiate. On continue avec la feuille suivante du dernier noeud ET :

$$\cos \frac{n + 1}{2} x = 0 \qquad (XII)$$

Sa solution est immédiate.
Il ne reste aucune branche de noeud ET non explorée dans l'arborescence, l'équation initiale est donc résolue.

L'arborescence de résolution développée est la suivante :



Exemple 4 :

$$\operatorname{tg} x + \operatorname{tg} 2x - \operatorname{tg} 3x = 0 \qquad \text{(I)}$$

Aucune règle obligatoire n'est applicable. Parmi les règles non obligatoires :

RNO a : $$\frac{\sin 3x}{\cos x \cos 2x} - \operatorname{tg} 3x = 0 \qquad \text{(II)}$$

Règles obligatoires :

RO i : $$\sin 3x - \cos x \cos 2x \operatorname{tg} 3x = 0 \qquad \text{(III)} \quad C = 7$$

La règle a est encore applicable de deux façon différentes à (I) :

RNO a : $$- \frac{\sin 2x}{\cos x \cos 3x} + \operatorname{tg} 2x = 0 \qquad \text{(IV)}$$

Règles obligatoires :

RO i : $$- \sin 2x + \cos x \cos 3x \operatorname{tg} 2x = 0 \qquad \text{(V)} \quad C = 7$$

RNO a : $$- \frac{\sin x}{\cos 2x \cos 3x} + \operatorname{tg} x = 0 \qquad \text{(VI)}$$

Règles obligatoires :

RO i : $$- \sin x + \cos 2x \cos 3x \operatorname{tg} x = 0 \qquad \text{(VII)} \quad C = 7$$

RNO g : $$\operatorname{tg} x + \frac{2 \operatorname{tg} x}{1 - \operatorname{tg}^2 x} - \frac{3 \operatorname{tg} x - \operatorname{tg}^3 x}{1 - 3 \operatorname{tg}^2 x} \qquad \text{(VIII)}$$

Règles obligatoires

RO i : $$2 \operatorname{tg}^5 x - 6 \operatorname{tg}^3 x = 0 \qquad \text{(IX)}$$

Mise en facteur :

$$2 \operatorname{tg}^3 x (\operatorname{tg}^2 x - 3) = 0 \qquad \text{(X)}$$

Le noeud (X) est marqué ET

On étudie : $$\operatorname{tg}^3 x = 0 \qquad \text{(XI)}$$

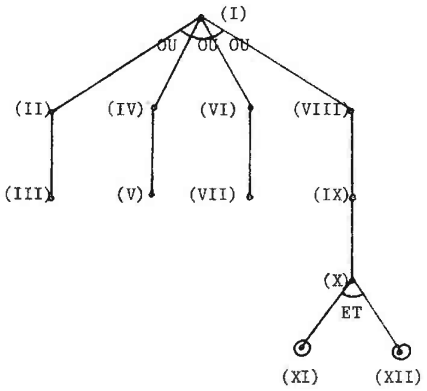
La solution est immédiate.

On étudie : $$\operatorname{tg}^2 x - 3 = 0 \qquad \text{(XII)}$$

La solution est immédiate.

Il n'existe pas de branche de noeud ET non traitée, l'équation initiale est donc résolue.

L'arborescence de résolution générée est la suivante :



Exemple 5 :

$$\sqrt{3} \operatorname{tg} x - 4 \sin^2 x = 0 \qquad \text{(I)}$$

Appliquons les règles obligatoires à (I) :

RO c : $$\sqrt{3} \frac{\sin x}{\cos x} - 4 \sin^2 x = 0 \qquad \text{(II)}$$

RO i : $$\sqrt{3} \sin x - 4 \sin^2 x \cos x = 0 \qquad \text{(III)}$$

Mise en facteur : $$\sin x (\sqrt{3} - 4 \sin x \cos x) = 0 \qquad \text{(IV)}$$

L'expression est sous forme d'un produit de facteurs, la première équation à résoudre est :

$$\sin x = 0 \qquad \text{(V)}$$

Sa solution est immédiate.

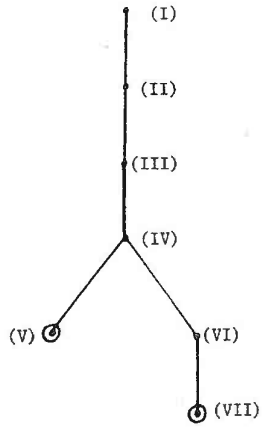
Il reste à résoudre l'équation :

$$- 4 \sin x \cos x + \sqrt{3} = 0 \qquad \text{(VI)}$$

RO d : $$\sqrt{3} \operatorname{tg}^2 x - 4 \operatorname{tg} x + \sqrt{3} = 0 \qquad \text{(VII)}$$

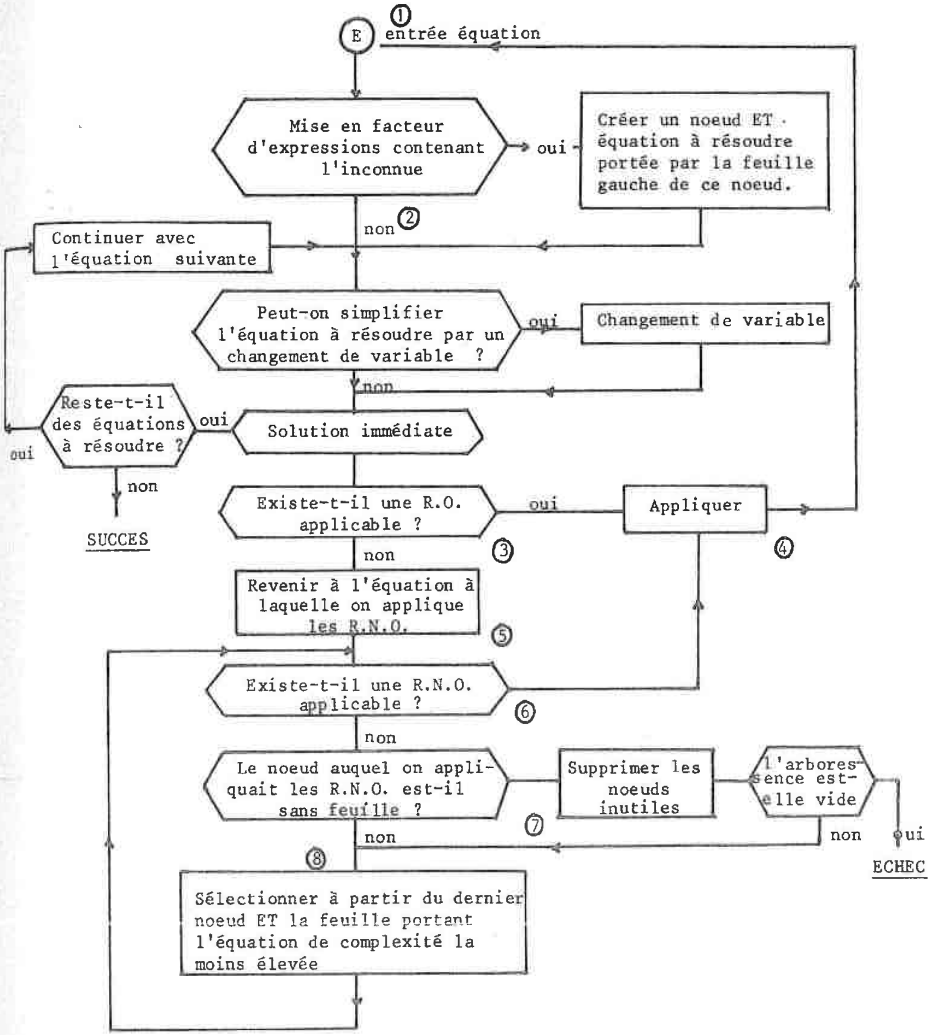
La solution de VII est immédiate. Il ne reste pas d'équation à résoudre, la résolution de (I) est donc achevée.

L'arborescence de résolution générée est :



ANNEXE A :

SCHEMA GENERAL DE FONCTIONNEMENT DU PROGRAMME



(cf. commentaires à la page suivante).

Commentaires de l'organigramme précédent

① Diverses initialisations sont effectuées en début de programme, notamment celles des pointeurs PEQ vers l'équation à traiter et PRNO vers la prochaine R.N.O. à appliquer.

② La mise en facteur d'éléments ne contenant pas l'inconnue intervient aussi à ce niveau, elle donne naissance à un nouveau noeud dont l'équation associée ne comporte plus ces facteurs. Certaines règles sont appliquées à l'expression avant et après la mise en facteur pour lui donner une "bonne" forme (cf. 4ème partie).

③ On essaie toujours d'appliquer les R.O. à partir de la première.

④ L'application d'une règle comporte toujours la simplification et la normalisation de l'expression obtenue.

⑤ Dans le cas où on étudie une nouvelle équation, c'est évidemment à cette équation que l'on va s'intéresser, il n'y a pas à revenir.

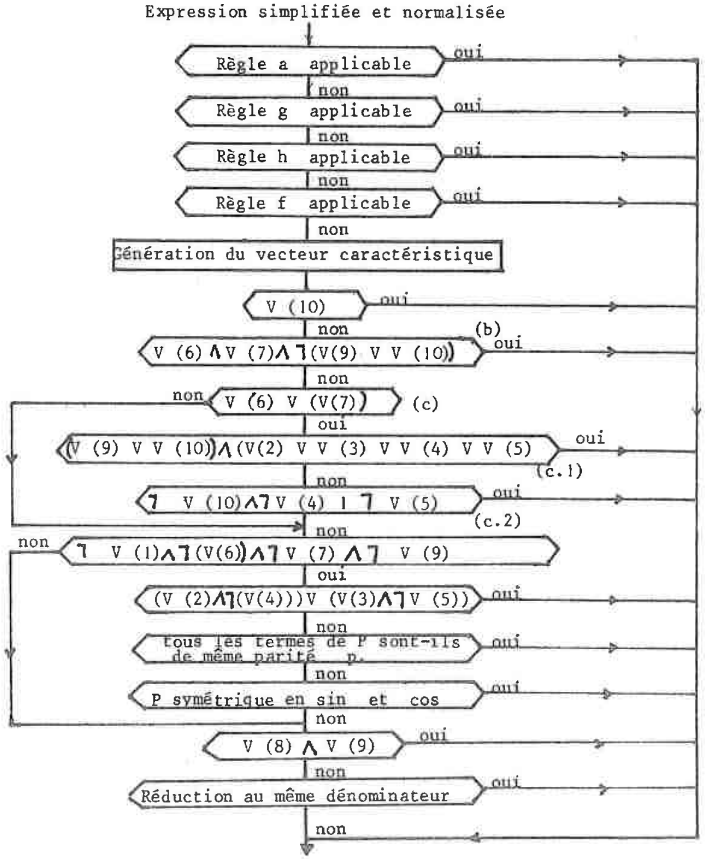
⑥ Lorsqu'on commence à appliquer des R.N.O. à un noeud, un compteur indique quelles sont les R.N.O. qu'on a déjà tenté de lui appliquer.

⑦ Il s'agit de supprimer dans l'arborescence le noeud auquel on ne peut appliquer aucune transformation et ceux qui dépendaient obligatoirement de lui.

⑧ Si plusieurs branches sont issues du noeud auquel on a appliqué des R.N.O., ce noeud est marqué OU.

ANNEXE B :

APPLICATION DES REGLES OBLIGATOIRES



Vecteur caractéristique pour la détermination des règles obligatoires

1	2	3	4	5	6	7	8	9	10
/	pair sin x	pair cos x	imp. sin x	imp. cos x	tg	cotg	x	2x	autre argument

V (I) = vrai si l'élément indiqué en I existe dans l'expression.

ANNEXE C :

LISTE DES GROUPES DE TRANSFORMATIONS TRIGONOMETRIQUES

Groupe I :

$$\sin u + \sin v \longrightarrow 2 \sin \frac{u+v}{2} \cos \frac{u-v}{2}$$

$$\sin u - \sin v \longrightarrow 2 \sin \frac{u-v}{2} \cos \frac{u+v}{2}$$

$$\cos u + \cos v \longrightarrow 2 \cos \frac{u+v}{2} \cos \frac{u-v}{2}$$

$$\cos u - \cos v \longrightarrow -2 \sin \frac{u+v}{2} \cos \frac{u-v}{2}$$

$$\operatorname{tg} u + \operatorname{tg} v \longrightarrow \frac{\sin (u+v)}{\cos u \cos v}$$

$$\operatorname{tg} u - \operatorname{tg} v \longrightarrow \frac{\sin (u-v)}{\cos u \cos v}$$

$$\operatorname{cotg} u + \operatorname{cotg} v \longrightarrow \frac{\sin (u+v)}{\sin u \sin v}$$

$$\operatorname{cotg} u - \operatorname{cotg} v \longrightarrow \frac{\sin (u-v)}{\sin u \sin v}$$

Groupe II :

$$\sin u \sin v \longrightarrow \frac{1}{2} (\cos (u-v) - \cos (u+v))$$

$$\cos u \cos v \longrightarrow \frac{1}{2} (\cos (u+v) + \cos (u-v))$$

$$\sin u \cos v \longrightarrow \frac{1}{2} (\sin (u+v) + \sin (u-v))$$

Groupe III :

$$\sin (u+v) \longrightarrow \sin u \cos v + \sin v \cos u$$

$$\cos (u+v) \longrightarrow \cos u \cos v - \sin u \sin v$$

$$\operatorname{tg} (u+v) \longrightarrow \frac{\operatorname{tg} u + \operatorname{tg} v}{1 - \operatorname{tg} u \operatorname{tg} v}$$

Groupe IV :

1. $\sin (-u) \longrightarrow -\sin u$
   $\cos (-u) \longrightarrow \cos u$
   $\operatorname{tg} (-u) \longrightarrow -\operatorname{tg} u$
   $\operatorname{cotg} (-u) \longrightarrow -\operatorname{cotg} u$

2. $\sin (\pi - u) \longrightarrow \sin u$
   $\cos (\pi - u) \longrightarrow -\cos u$
   $\operatorname{tg} (\pi - u) \longrightarrow -\operatorname{tg} u$
   $\operatorname{cotg} (\pi - u) \longrightarrow -\operatorname{cotg} u$

3. $\sin (\pi + u) \longrightarrow -\sin u$
   $\cos (\pi + u) \longrightarrow -\cos u$
   $\operatorname{tg} (\pi + u) \longrightarrow \operatorname{tg} u$
   $\operatorname{cotg} (\pi + u) \longrightarrow \operatorname{cotg} u$

4. $\sin (\frac{\pi}{2} - u) \longrightarrow \cos u$
   $\cos (\frac{\pi}{2} - u) \longrightarrow \sin u$
   $\operatorname{tg} (\frac{\pi}{2} - u) \longrightarrow \operatorname{cotg} u$
   $\operatorname{cotg} (\frac{\pi}{2} - u) \longrightarrow \operatorname{tg} u$

5. $\sin (\frac{\pi}{2} + u) \longrightarrow \cos u$
   $\cos (\frac{\pi}{2} + u) \longrightarrow -\sin u$
   $\operatorname{tg} (\frac{\pi}{2} + u) \longrightarrow -\operatorname{cotg} u$
   $\operatorname{cotg} (\frac{\pi}{2} + u) \longrightarrow -\operatorname{tg} u$

Groupe V :

$$\sin (\frac{\pi}{4} + u) \longrightarrow \frac{\sqrt{2}}{2} (\sin u + \cos u)$$

$$\cos (\frac{\pi}{4} + u) \longrightarrow \frac{\sqrt{2}}{2} (\cos u - \sin u)$$

Groupe VI :

1. $\sin 2u \longrightarrow 2 \sin u \cos u$

   $\cos 2u \longrightarrow 2\cos^2 u - 1$

2. $\operatorname{tg} 2u \longrightarrow \dfrac{2 \operatorname{tg} u}{1 - \operatorname{tg}^2 u}$

3. $\sin 3u \longrightarrow 3 \sin u - 4 \sin^3 u$

   $\cos 3u \longrightarrow 4 \cos^3 u - 3 \cos u$

   $\operatorname{tg} 3u \longrightarrow \dfrac{3 \operatorname{tg} u - \operatorname{tg}^3 u}{1 - 3 \operatorname{tg}^2 u}$

Groupe VII :

1. $\sin^2 u \longrightarrow \dfrac{1 + \cos 2u}{2}$

   $\cos^2 u \longrightarrow \dfrac{1 - \cos 2u}{2}$

   $\operatorname{tg}^2 u \longrightarrow \dfrac{1 - \cos 2u}{1 + \cos 2u}$

2. $\sin^3 u \longrightarrow \dfrac{1}{4}(3 \sin u - \sin 3u)$

   $\cos^3 u \longrightarrow \dfrac{1}{4}(\cos 3u + 3 \cos u)$

Groupe VIII :

$\sin 2u \longrightarrow \dfrac{2 \operatorname{tg} u}{1 + \operatorname{tg}^2 u}$

$\cos 2u \longrightarrow \dfrac{1 - \operatorname{tg}^2 u}{1 + \operatorname{tg}^2 u}$

$\operatorname{tg} 2u \longrightarrow \dfrac{2 \operatorname{tg} u}{1 - \operatorname{tg}^2 u}$

Groupe IX :

$\operatorname{tg} u \longrightarrow \dfrac{\sin u}{\cos u}$

$\operatorname{cotg} u \longrightarrow \dfrac{\cos u}{\sin u}$

Groupe X :

$\operatorname{cotg} u \longrightarrow \dfrac{1}{\operatorname{tg} u}$

Groupe XI : $\sin^2 u \longrightarrow 1 - \cos^2 u$

Groupe XII : $\cos^2 u \longrightarrow 1 - \sin^2 u$

Groupe XIII : $\sin u \cos u \longrightarrow \dfrac{1}{2} \sin 2u$

Groupe XIV : $\cos^2 u \longrightarrow \dfrac{1}{1 + \operatorname{tg}^2 u}$

Groupe XV :

$\dfrac{\sin u}{\cos v} \longrightarrow \operatorname{tg} u$

$\dfrac{\cos u}{\sin v} \longrightarrow \dfrac{1}{\operatorname{tg} u}$

ANNEXE D :

TABLE DE CONNEXION ENTRE "BUTS" A ATTEINDRE ET TRANSFORMATIONS SUSCEPTIBLES DE PERMETTRE D'ATTEINDRE CES BUTS

Groupe \ But	Obtenir une équation de la forme :				Linéariser	Ramener les arguments des fonctions trigonométriques à x
	F(cos2x)= 0	F(cos x)= 0	F(sinx)= 0	F(tgx)= 0		
VI - 1		x	x			x
VI - 2						x
VI - 3						x
VII - 1	x				x	
VII - 2					x	
VIII				x		
IX						
X					x	x
XI	x	x			x	
XII			x			
XII					x	
XIV				x		
XV				x		

Dans chaque colonne, les croix indiquent les groupes de transformations que l'on essaie d'appliquer à l'expression étudiée.
Comme pour toute application de règles, ces transformations sont appliquées en tout point de l'expression jusqu'à ce qu'aucune d'entre elles ne soit plus applicable.

4ème partie

NOTATIONS - CALCUL FORMEL
RESULTATS

Nous aborderons dans cette partie les problèmes "techniques" soulevés par la réalisation matérielle du programme P.R.E.T.

Ces problèmes portent essentiellement sur la représentation des symboles algébriques et trigonométriques et des expressions formées de ces symboles, sur la transformation de ces expressions et sur les formes "simples" ou "normales" auxquelles on souhaite aboutir par transformations successives. Nous allons donc développer ces différents points et indiquer les solutions retenues.

I - REPRESENTATION DES SYMBOLES ET DES EXPRESSIONS

1) - Les notations possibles :

- La notation infixée :

Le mathématicien utilise sur sa feuille de papier une notation classique dite "infixée" pour les expressions trigonométriques qu'il manipule.
Par exemple, il écrit :

sin x + 2 (tg 2x + cos 2x)

Mais il peut aussi avoir les formes suivantes :

sin x + 2 tg 2x + 2 cos 2x

ou encore

(sin x + 2 tg 2x) + cos 2x + cos 2x

ou encore

sin x + 2 cos 2x + 2 tg 2x

Si l'expression reste relativement courte, il verra au premier coup d'oeil qu'il est en présence de quatre "écritures" différentes pour une même expression.
Cette notation infixée n'est donc pas sans ambiguité. De plus, elle ne permet ni d'accéder de façon systématique à certains éléments de l'expression ni de mettre en évidence la structure de cette expression. En machine il faut choisir une représentation qui permette de remédier à ces inconvénients.
Toutes les ambiguités présentes sur cet exemple ne sont pas de même nature, une meilleure notation ne fera évidemment pas disparaître celles qui sont

liées à la commutativité ou à l'associativité de certains opérateurs, elle doit cependant permettre des traitements systématiques.

Deux types de notations répondent à ces exigences :

- Des notations linéaires (préfixée et suffixée) :
Dans la notation polonaise préfixée, on écrit l'opérateur suivi de son (ou ses) opérande (s). Par exemple :

+ x y est interprété comme x plus y
- + x y z est interprété comme (x plus y) moins z
+ x neg z est interprété comme x plus (moins z)
(neg représente l'opérateur unaire "moins").

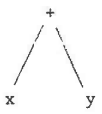
Cette notation a l'avantage de ne pas nécessiter l'introduction de parenthèse. On pourra donc représenter des expressions en utilisant un minimum d'espace mémoire. Par ailleurs, il existe sur cette notation un théorème fondamental qui permet de découper facilement le terme commençant à un endroit donné. Elle a été utilisée par VIVET dans son programme.

Il existe aussi une notation suffixée où l'on donne d'abord les opérances puis l'opérateur correspondant. Par exemple :

x y + est interprété comme x + y
x y + tg " " tg (x + y)

Cette notation présente des avantages analogues à la précédente. Elle est plus rapide lorsqu'il s'agit d'évaluer des expressions arithmétiques. Elle a été utilisée par LAURENT dans DALI.

- Des notations en arborescence :
Il en existe de nombreux types. La plus naturelle consiste à faire correspondre à un opérateur ses opérandes selon le schéma suivant :



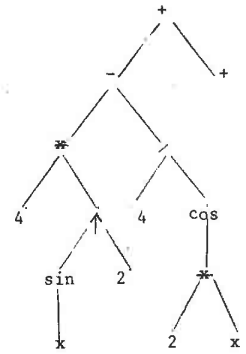
pour x + y

pour "moins" x

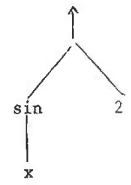
La structure de l'expression représentée est bien mise en évidence. Si l'on veut revenir à une notation infixée, il suffit de projeter les symboles figurant aux noeuds de l'arborescence sur un axe horizontal en complétant avec des parenthèses. Si l'on veut apporter des modifications à certaines parties de l'expression, il suffit de remplacer les sous-arborescences correspondantes par d'autres. Par exemple :

$$4 \sin^2 x - \frac{4}{\cos 2x} + 7$$

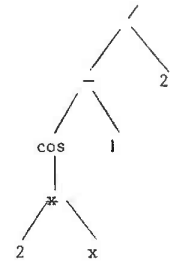
peut être représenté par :



Pour effectuer la transformation $\sin^2 x \longrightarrow \frac{\cos 2x - 1}{2}$, on remplacera



par



Il est donc facile d'effectuer des modifications avec de telles représentations. Mais elles demandent évidemment un espace mémoire plus grand et l'utilisation de langages de liste ou d'un outillage de liste adapté à un langage comme FORTRAN pour la programmation. Beaucoup de programmes utilisent des représentations en listes : celui de SLAGLE est écrit en LISP, celui de JONCOUR en LEAP un langage associatif dérivé d'ALGOL 60 ; DATAL comporte un outillage écrit en ALGOL 60 pour travailler sur des listes, les programmes de VIVET et LAURENT en FORTRAN utilisent partiellement ce type de représentation.

2) - Résultats fondamentaux sur la notation polonaise :

On trouvera davantage de détails et une référence à un ouvrage de logique dans (38). Nous reprenons simplement quelques définition et théorèmes.

. Rang d'une suite de symboles :

- Si la suite de symboles est vide, son rang est 0.
- Si elle se réduit à une variable, son rang est - 1.
- Si elle se réduit à une connective n-aire, son rang est n - 1.
- Si la suite de symboles est formée par la concaténation de la suite de symboles A et du symbole S, son rang est r (A) + r (S).

. Théorème fondamental :

La suite de symboles A est un terme si et seulement si :

1) r (A) = - 1

2) Pour toute suite B qui est une tête propre de A ($\exists$ C, C non vide, tel que A = BC), r (B) doit être positif ou nul.

. Exemples sur des expressions arithmétiques :

Les symboles à prendre en considération sont les constantes numériques ou formelles, les variables, l'opérateur unaire - (noté neg), les opérateurs binaires +, -, x, /, ↑ (↑ représente l'élévation à une puissance) et les lignes trigonométriques sin, cos, tg, cotg qui se comportent ici comme des opérateurs unaires.

rang (+ sin + x neg y 1 2) = 1 + 0 + 1 + 0 - 1 + 0 - 1 - 1 - 1 = - 2

rang (+ * sin x cos y * sin y cos x) = 1 + 1 + 0 - 1 + 0 - 1 + 1 + 0 - 1
+ 0 - 1 = - 1

Le calcul du rang de ces deux expressions nous a permis de vérifier que la première n'est pas bien formée, alors que la seconde est bien un terme. Le théorème fondamental permet de déterminer rapidement où se termine un terme qui commence à un endroit déterminé de l'expression considérée ; il suffit de calculer les rangs des suites de symboles rencontrés à partir de cet endroit et de s'arrêter dès que la valeur du rang atteint - 1.

Exemple : + a + / + ↑ tg x 2 1 + ↑ tg x 2 neg 1 tg x

Regardons avec cette méthode où se termine le terme commençant au quatrième symbole de l'expression.

Valeurs successives du rang :

1 2 3 3 2 1 0 1 2 2 1 0 0 - 1

Ce terme se termine au 17ème symbole.
De tels calculs interviennent constamment dans l'algorithme d'unification que nous présentons maintenant.

3) - L'algorithme d'unification :

Lorsqu'on veut transformer l'une des expressions que notre programme manipule, on essaie de lui appliquer une règle, par exemple la règle tg x ⟶ $\frac{\sin x}{\cos x}$.

Mais il est rare qu'on puisse le faire directement. En effet, si l'expression considérée est sina + tg (x + a), il faudra d'abord effectuer dans les deux membres de la règle la substitution x ⟶ x + a avant de "reconnaître" l'identité entre un terme de l'expression et le premier membre de la règle. On pourra alors remplacer le premier membre reconnu par le second modifié par les substitutions.

On trouvera dans (38) la justification et l'exposé d'un algorithme qui donne systématiquement les substitutions amenant le résultat le plus général possible. Il est appliqué à des démonstrations de théorèmes en logique.

Nous en rappelons le principe pour expliquer comment on peut l'utiliser de façon systématique pour appliquer à une expression toutes les règles algébriques ou trigonométriques que l'on souhaite.

Algorithme :

Soit l'objet E auquel on veut appliquer la règle A ⟶ B.

(A, B et E sont des suites de symboles en notation polonaise)

E : $e_1, e_2, \ldots\ldots\ldots\ldots, e_n$

A : $a_1, a_2, \ldots\ldots\ldots\ldots, a_p$

B : $b_1, b_2, \ldots\ldots\ldots\ldots, b_q$

Le but est d'amener E et A à coïncider en ne faisant que les substitutions indispensables.

1) On vérifie qu'aucune variable ne figure à la fois dans E et A (sinon il faut substituer à toutes les occurrences de cette variable dans E une autre variable qui ne figure ni dans E, ni dans A, ni dans B).

2) En supposant que les m premiers symboles de E et A coïncident, on compare e_{m+1} et a_{m+1} ; trois cas peuvent se présenter :

. e_{m+1} et a_{m+1} sont des connectives :

Si les connectives sont les mêmes, il y a coïncidence de m + 1 symboles, on continue en 2) avec m : = m + 1. Sinon il y a échec.

. e_{m+1} et a_{m+1} sont des variables :

Dans ce cas, on remplace chaque occurrence de a_{m+1} par e_{m+1} dans

e_{m+2}, ..., en

a_{m+2}, ..., ap

b_1, ..., bq

Il y a alors coïncidence de m + 1 symboles. On continue en 2) avec m : = m + 1.

. a_{m+1} est une variable et e_{m+1} une connective (ou l'inverse).

On remplace alors chaque occurrence de a_{m+1} dans

e_{m+2}, ..., en

a_{m+2}, ..., ap

b_1, ..., bq

par le terme qui commence en e_{m+1}, (ce terme est déterminé en utilisant le théorème fondamental précédent) à condition que la variable a_{m+1} n'apparaisse pas dans e_{m+2} ..... e_n.

Si cette condition est réalisée, soit k le nombre de symboles du terme qui commence en e_{m+1}, on poursuit en comparant e_{m+1+k} avec a_{p+2}.

On poursuit jusqu'à ce que les chaînes A et E soient épuisées (elles le seront en même temps à cause des propriétés de la notation polonaise).

3) En cas de succès, remplacer l'expression E par la nouvelle forme de B.

Application :

Soit à utiliser la règle sin (a + b) ⟶ sin a cos b + sin b cos a sur l'expression sin (2 x + α).

Reprenons la méthode précédente :

E : SIN + * 2 x α
A : SIN + a b
B : + * SIN a cos b * SIN b cos a

Les deux premiers symboles de E et A coïncident. Au troisième symbole, nous remplaçons dans E, A, B a par * 2 x, ce qui donne :

E : SIN + * 2 x α
A : SIN + * 2 x b
B : + * SIN * 2 x COS b * SIN b COS * 2 x

Au 6ème symbole de E, nous sommes amenés à remplacer partout b par α, ce qui donne :

E : SIN + * 2 x α
A : SIN + * 2 x α
B : + * SIN * 2 x COS α * SIN α COS * 2 x

L'application de la règle consiste à remplacer E par la dernière forme de B.

4) - Les solutions adoptées dans P.R.E.T. :

Les symboles

Les symboles utilisés sont les suivants :

constantes entières
variables (inconnue, paramètres formels, variables formelles des règles)
connectives unaires : - (représenté par NEG), $\sqrt{\ }$ (représenté par RAC)
sin, cos, tg, cotg
connectives binaires : +, *, /, ↑ (puissance représentée par **).

Les constantes sont représentées par elles-mêmes, les autres symboles sont codés par des entiers supérieurs à 5 000. Le "moins" binaire a été remplacé par + et NEG ; a - b est écrit : + a neg b.

La représentation des expressions

La notation retenue est la notation polonaise préfixée.
Nous avons indiqué quels étaient ses avantages. Elle présente des inconvénients lorsqu'on veut réarranger les termes d'une expression, avec une notation en liste il suffit de modifier des pointeurs alors que nous sommes obligés de recopier l'expression. Certains auteurs (LAURENT, VIVET) ont adopté des représentations mixtes, le programme dépense alors du temps à passer d'une représentation à l'autre. Chaque solution a ses avantages, on ne peut les classer que selon un critère précis (langage dont on dispose, place mémoire, etc...), pas dans l'absolu.

L'application des règles

Les règles sont appliquées aux différentes expressions engendrées à l'aide de l'algorithme d'unification. Pour éviter des tests supplémentaires, les variables figurant dans les règles ne sont jamais les mêmes que celles qui peuvent figurer dans les équations.

Exemple : on aura à unifier sin (u + v) $\longrightarrow$ sin u cos v + sin v cos u (règle) avec sin (x + a) terme d'une équation.

L'algorithme décrit précédemment a été programmé en tenant compte de ces restrictions. Il n'y a pas à vérifier que le premier membre de la règle et l'équation n'ont pas de variable commune ; et au cours de l'unification lorsqu'on se trouve en comparant deux symboles dans le troisième cas indiqué, on aura toujours une variable dans le premier membre de la règle.

En plus de l'ensemble des règles algébriques ou trigonométriques fournies au programme, il nous a semblé commode de représenter sous forme de "règles" certaines opérations.

Par exemple, pour effectuer les calculs sur les constantes numériques, il est commode de regrouper ces constantes à la fin du terme qui les contient et de les remplacer ensuite par le résultat du calcul. Nous pouvons utiliser l'algorithme d'unification en le complétant pour lui permettre d'accepter des "règles conditionnelles" :

$$+\ NE\ NN \longrightarrow +\ NN\ NE$$
$$*\ NE\ NN \longrightarrow *\ NN\ NE$$

où NE désigne une constante numérique et NN n'importe quelle expression qui n'est pas une constante numérique. Avant de faire effectivement une transformation, le programme vérifie que les termes trouvés pour NE et NN appartiennent bien aux catégories "constante numérique" ou "différent de constante numérique", le programme distingue ces règles par les codes des symboles qui y figurent :

- dans les règles qui s'appliquent systématiquement les variables formelles sont toujours U, V, W, T.
- dans les règles conditionnelles les variables formelles ne sont jamais U, V, W, T.

Exemple : appliquons ces règles à

$$*\ 2\ *\ \cos x\ *\ 2\ *\ \sin x \cos x$$

on obtient :

$$*\ \cos x\ *\ \sin x\ *\ \cos x\ *\ 2\ 2$$

Il nous a semblé intéressant d'achever le calcul par le même procédé, nous avons donc prévu des règles :

$$+\ NE\ NF \longrightarrow 0$$
$$*\ NE\ NF \longrightarrow 0$$
$$/\ NE\ NF \longrightarrow 0$$
$$\uparrow\ NE\ NF \longrightarrow 0$$

où NE et NF désignent une constante numérique.

Le deuxième membre de chacune de ces règles est initialisé à 0. Sa valeur exacte est calculée au moment de l'unification. Dans l'exemple précédent, la deuxième règle s'applique à

$$*\ 2\ 2.$$

La multiplication des deux constantes est alors effectuée et son résultat placé dans le deuxième membre de la règle, soit ici

$$*\ 2\ 2 \longrightarrow 4$$

L'unification s'achève en utilisant ce résultat.

Enfin le programme est parfois amené à faire des changements de variables, par exemple changer x en $x+\frac{\pi}{4}$. Ces changements sont également donnés sous la forme de règles et effectués avec l'algorithme d'unification.

II - CALCUL FORMEL SUR LES EXPRESSIONS

Dans la troisième partie, nous avons énoncé un certain nombre de règles que le programme applique dans certaines circonstances. Mais il y a évidemment bien d'autres règles à appliquer, nous venons d'en voir un exemple avec le calcul sur les constantes numériques. L'objet de ce chapitre est donc de voir quelles règles appliquer, à quel moment et selon quels critères.

1) - Nature du problème :

Exemple 1 :

Considérons l'équation cos 5x - cos 3x + sin x = 0.
L'application des règles non obligatoires nous amène à unifier un terme de l'expression avec la règle cos p - cos q $\longrightarrow$ $-2 \sin \frac{p+q}{2} \sin \frac{p-q}{2}$

Le terme + COS * 5x NEG COS * 3x
va être réécrit :

NEG * 2 * SIN / + * 5x * 3x 2 SIN / + * 5x NEG * 3x 2

On ne conserve pas en général des expressions du type :

$$-2 \sin \frac{5x+3x}{2} \quad \sin \frac{5x-3x}{2}$$

Les transformations nécessaires, on dit souvent "simplifications", sont le regroupement des termes semblables, ici les termes en x et la division du coefficient de x par 2 qu'on peut considérer comme une simplification de fraction si l'on présente l'argument du premier sinus sous les formes :

/ * 2 * 4x 2 ou / * 2 * 2 * 2x 2

ou comme un simple calcul numérique si on l'écrit :

* / 8 2x

Exemple 2 :

Considérons l'équation 4 sin 3x + cos 2x - sin x + 1 = 0.
L'application des règles non obligatoires nous amène à transformer les termes de l'expression qui présente un argument autre que x sous un symbole trigonométrique.

Nous appliquons les deux règles suivantes l'une après l'autre :

SIN * 3 U $\longrightarrow$ + NEG * 4 ↑ SIN U 3 * 3 SIN U

COS * 2 U $\longrightarrow$ + * 2 ↑ COS U 2 NEG 1

Ce qui donne l'expression
+ * 4 + NEG * 4 ↑ SIN x 3 * 3 SIN x + + * 2 ↑ COS x 2 NEG 1 + NEG SIN x 1

Il faut ici définir quelles transformations nous utiliserons pour "simplifier" cette expression, c'est-à-dire effectuer le calcul sur les constantes + 1 et - 1 et regrouper les deux termes en sin x qui y figurent. Il faudra donc sortir le terme * 3 SIN x du facteur où il se trouve, c'est-à-dire appliquer une règle du type :

* U + V W $\longrightarrow$ + * UV * U W

Ces deux exemples donnent une idée de la diversité et de la complexité des représentations devant lesquelles le programme se trouve après l'application des formules de trigonométrie. Il est donc nécessaire d'adopter une politique de simplification systématique des expressions obtenues et d'abord de définir ce que l'on entend par "expression simple".

2) - Politique de simplification :

Tous les auteurs de programmes de calcul formel ont dû adopter des critères de simplification. En 1971, MOSES (10) a fait une excellente synthèse des travaux connus et a dégagé un certain nombre de types.

La première remarque que l'on puisse faire est qu'il n'existe pas de simplification dans l'absolu. Une simplification, c'est une transformation qui appliquée à une expression algébrique permet d'obtenir une nouvelle forme de celle-ci à partir de laquelle le calcul sera poursuivi de façon efficace. Tout dépend donc du calcul que l'on a à faire, c'est-à-dire du contexte.

Une expression peut apparaître simple dans un contexte et compliquée dans un autre. Si on s'attache au nombre des symboles

$\frac{x^7}{x^{12}+1}$ paraît plus simple que $\frac{\frac{1}{4}(4x^3)\,x^4}{(x^4)^3+1}$

Cependant, dans un programme d'intégration formelle, on ne ramènera pas la forme 2 à la forme 1, parce que la forme 2 conduira à une solution.

Certaines transformations semblent s'imposer dans tous les cas.
Par exemple : $U + 0 \longrightarrow U$, $U \uparrow 1 \longrightarrow U$, $U * 1 \longrightarrow U$.
Cependant dans une expression de la forme $\sin x * 1 + \sin x \cos^2 x$ la mise en facteur de $\sin x$ par application d'une règle comme $U * A + U * B \longrightarrow U * (A + B)$ devient impossible si on supprime le terme $\sin x * 1$. Il faudra soit pouvoir faire réapparaître le 1, soit ne pas effectuer les mises en facteur par application de la règle précédente... à moins que dans un contexte bien particulier on ne fasse jamais de mise en facteur.

Voyons donc quels sont les systèmes de simplification possibles. Les qualificatifs utilisés sont les traductions de ceux employés par MOSES dans son article.

- Les systèmes "radicaux" ramènent toute expression à une forme canonique bien définie. Dans ces systèmes, deux expressions équivalentes ont exactement la même représentation. Certaines améliorations locales ont été apportées, par exemple l'interdiction de développer $(x + 1)^n$ sous forme de polynome à partir d'une certaine valeur de n ; elles ont conduit aux systèmes "nouvelle gauche".

- Les systèmes "libéraux", à l'inverse des précédents ne font que certaines simplifications au fur et à mesure de la demande et de façon locale dans l'expression. Ils se rapprochent du comportement humain mais du point de vue programmation ils sont peu efficaces car coûteux en espace et en temps. Les systèmes "conservateurs" opposés des radicaux ne considèrent aucune simplification comme systématique.

- Les systèmes "catholiques" essaient d'avoir les avantages de chacune des catégories précédentes tout en limitant les défauts. Certaines simplifications sont effectuées de façon "radicale" sur des formes canoniques, mais d'autres représentations seront conservées.

Parmi les programmes que nous avons décrits au chapitre 2, celui de JONCOUR par l'utilisation de formes canoniques a adopté une politique assez "radicale" de simplification, alors que VIVET et LAURENT se classent dans les systèmes "catholiques".

3) - <u>La simplification dans la résolution d'équations trigonométriques</u> :

Quelle politique de simplification fallait-il adopter dans PRET ? Nous nous sommes heurtés à beaucoup de difficultés dans ce domaine, étant donné le nombre d'impératifs auxquels il fallait se soumettre. Les expressions manipulées peuvent être longues, approchant parfois 50 symboles, il fallait donc systématiser au maximum ces opérations, c'est-à-dire définir un certain nombre de transformations "radicales".

Par ailleurs, nous ne pouvions pas admettre n'importe quelle "forme" pour nos expressions une fois les calculs intermédiaires effectués. La plupart de nos heuristiques sont directement ou indirectement fondées sur la forme de l'équation à résoudre : il est commode par exemple pour l'application de certaines règles obligatoires de reconnaître rapidement un sinus à une puissance paire, cela nous entraîne à préférer certaines représentations à d'autres :

exemple : $\sin^4 x \cos x$ et $(\sin^2 x)^2 \cos x$
sont préférables à
$$\sin x * \sin x * \sin x * \sin x * \cos x$$

Considérons maintenant l'expression :
$$\cos x (\sin x + 1) + \cos 3x.$$
Parmi les règles trigonométriques que nous avons données, seule la transformation de $\cos 3x$ est applicable sur cette forme. Si nous l'écrivons maintenant :
$$\cos x \sin x + \cos x + \cos 3x$$
nous pouvons unifier le terme $\sin x \cos x$ avec $\sin u \cos u \longrightarrow \frac{1}{2} \sin 2u$
et le terme
$$\cos x + \cos 3x \text{ avec } \cos u + \cos v \longrightarrow 2 \cos \frac{u+v}{2} \cos \frac{u-v}{2}$$

Nous avons donc été amenés à orienter ces transformations radicales en fonction de la forme que nous souhaitions voir le plus souvent apparaître pour appliquer commodément nos heuristiques.

Calculs algébriques "immédiats" :

Nous appliquons systématiquement les règles suivantes qui font intervenir les constantes 0 et 1 :

+	U	0
+	0	U
*	0	U
*	U	0
**	U	0
**	0	U
/	0	U
COS	0	
SIN	0	
TG	0	

*	1	U
*	U	1
/	U	1
**	U	1
**	1	U

N. B. : (** représente l'opérateur puissance).

Nous effectuons aussi systématiquement les calculs (somme, produit, quotient s'il est exact, élévation à une puissance) portant sur des constantes numériques.

Nous appliquons trois transformations trigonométriques qui réduisent le nombre de termes d'une expression :

$\cos^2 u + \sin^2 u \longrightarrow 1$

$1 - \sin^2 u \longrightarrow \cos^2 u$

$1 - \cos^2 u \longrightarrow \sin^2 u$

Enfin, les lignes trigonométriques de - u sont écrites en fonction de celles de u (sin (- x) ⟶ - sin x, etc...)

Forme simplifiée d'une expression :

La plupart des autres transformations possibles sont liées aux propriétés de commutativité, associativité de certains opérateurs et de distributivité des opérateurs les uns par rapport aux autres.

Opérateur NEG.

Outre certaines simplifications comme NEG NEG U ⟶ U et ↑ NEG U 2 ⟶ ↑U 2, + U NEG U ⟶ 0, *NEG U ↑ ⟶ 0, nous appliquons des règles destinées à placer NEG devant les connectives *, /, ↑ soit :

* U NEG V ⟶ NEG * U V
* NEG U V ⟶ NEG * U V
/ U NEG V ⟶ NEG / U V
/ NEG U V ⟶ NEG / U V
↑ NEG U N ⟶ NEG ↑ U N (N impair, règle conditionnelle)

Par contre lorsqu'une connective + existe dans un terme nous la gardons comme connective principale, ce qui implique la règle :

NEG + U V ⟶ + NEG U NEG V

Ceci facilité le regroupement des termes semblables.

Opérateurs + et *.

Nous appliquons systématiquement la distributivité de * par rapport à +

* U + V W ⟶ + * U V * U W
* + U V W ⟶ + * U W * V W

Opérateurs + et ↑.

Les sommes des termes élevées à des puissances sont développées :

↑ + U V 2 ⟶ + ↑U 2 + ↑V 2 * 2 * U V
↑ + U V 3 ⟶ + ↑U 3 + ↑V 3 + * 3 * ↑U 2 V * 3 * ↑V 2 U

Opérateurs * et /.

Les règles qui amènent / en tête du terme considéré sont appliquées :

```
* U / V W  ------------>  / * U V * U W
* / U V W  ------------>  / * U W * V W
/ U / V W  ------------>  / * U W V
// U V W  ------------>  / U * V W
```

Opérateurs *, / et ↑

La connective * est ramenée en tête du terme ;

```
↑ * U V W  ------------>  * ↑U W ↑V W
↑ / U V W  ------------>  / ↑U W ↑V W
```

Calcul sur les puissances:

```
* U U  ------------>  ↑ U 2
* ↑U V U  ------------>  ↑ U + V 1
* ↑U V  U W  ------------>  ↑ U + V W
↑↑U V W  ------------>  ↑ U * V W
```

En résumé, l'ensemble de ces règles tend à donner à l'expression ainsi simplifiée la forme d'une somme de termes, chaque terme de cette somme est un produit ou un quotient éventuellement précédé de l'opérateur unaire NEG.

Exemple :

cos x sin x cos x + 2 (- sin 2x) + 7 - 3

Cette opération sera simplifiée en :

$\cos^2 x \sin x + - 2 \sin 2x + 4$

Exemple :

$$\left(\frac{\sin x + 1}{\sin x}\right)^2 + 2 * \frac{1}{\sin x}$$

Cette équation sera transformée en :

$$\frac{\sin^2 x + 2 \sin x + 1}{\sin^2 x} + \frac{2}{\sin x}$$

Ordre d'application des règles :

L'ordre d'application des règles que nous venons d'énoncer n'est pas indifférent. Il pose peu de problèmes lorsque l'expression ne comporte pas de fractions, dans le cas de fractions il faut prévoir un traitement spécial

local à la fraction :

Exemple : Soit à appliquer les formules de passage en tangente de l'arc moitié à l'expression

$$1 + \frac{\sin 2x}{\sin^2 2x + 1}$$

Nous obtenons :

$$1 + \frac{\frac{2 \operatorname{tg} x}{1 + \operatorname{tg}^2 x}}{\left(\frac{2 \operatorname{tg} x}{1 + \operatorname{tg}^2 x}\right)^2 + 1}$$

Il serait évidemment maladroit de développer $(1 + \operatorname{tg}^2 x)^2$, cela empêcherait une simplification qui ne réapparaîtrait pas ensuite.

Recherche d'un facteur commun :

Nous avons parlé dans la troisième partie du sous-programme de mise en facteur en précisant que l'expression à traiter devait être sous une "bonne forme". Pour chercher un facteur commun, il faut évidemment que les facteurs de chaque terme apparaissent facilement, nous sommes donc conduits à écrire les puissances sous forme de produits et les constantes entières sous forme de produits de facteurs premiers.

A la sortie du sous-programme de mise en facteurs il faut bien sûr opérer les transformations inverses.

Commutativité des opérateurs + et * :

Dans les règles précédemment indiquées rien n'indique si nous devons garder sin x + cos x + 1 ou cos x + sin x + 1 comme représentation d'expression. Ce problème est important puisque nous sommes amenés à comparer des expressions entre elles. La même question se pose pour cos 2 x * sin x et sin x * cos 2x.

Pour lever la plupart de ces ambiguités, nous définissons un code pour chaque terme, dans tout ce qui suit l'expression est une somme de termes, chaque terme est un produit ou quotient de "facteurs".

code (terme) = $\sum$ code (facteurs non constants)

Exemple : code (2 cos x) = code (cos x) = code (cos) + code (x)

Les codes choisis sont ceux qui nous ont permis de coder nos symboles dans le programme.
Un produit de facteurs est écrit par ordre de codes décroissants des facteurs

si code (cos 2 x) > code (sin x) sin x cos 2 x s'écrit cos 2x sin x

Une expression est écrite par ordre de codes décroissants des termes :

si code (cos 2x sin x) > code (sin x) > code (1)

1 + cos 2x sin x - sin x s'écrit cos 2x sin x - sin x + 1.

Ce classement est évidemment de nature heuristique. Certaines ambiguités demeurent si 2 termes ou 2 facteurs ont même code par suite de compensations dans les calculs de code. Ceci est très rare. Il faut noter de plus que le fait de ne pas reconnaître deux expressions égales ne nous amènera jamais à des calculs faux, il faut seulement nous éloigner de la solution.

Associativité des opérateurs + et * :

L'associativité de + nous permet d'écrire

a + (b + c) + d ou (a + b) + (c + d) ou a + b + (c + d)

ce qui donne en notation polonaise :

+ a + + b c d
+ + a b + c d
+ a + b + c d

Il faut évidemment ne retenir que l'une des formes possibles. Pour cela notre programme applique systématiquement les règles :

+ + U V W ⟶ + U + V W
* * U V W ⟶ * U * V W

Notons en conclusion que toutes les transformations dont nous venons de parler ne changent pas l'équation elle-même, elles ne visent qu'à l'écrire sous une autre forme. On peut cependant penser à d'autres règles qui simplifient la résolution, nous les regroupons sous le nom de normalisation.

4) - Normalisation des équations :

Une première normalisation consiste à réduire au même dénominateur tous les termes de l'expression et à chasser ce dénominateur en le supposant non nul. Elle intervient par programme au niveau des règles obligatoires.

Un deuxième type de normalisation de ce type concerne les changements de variables.

Exemple : sin 4x + sin 2x - 1 = 0

Cette équation est plus simple si nous l'écrivons :

sin 2x + sin x - 1 = 0

avec le changement de variable 2 x ⟶ x. Il est difficile de déceler tous les changements de variables possibles. Cependant, le changement 2x ⟶ x intervient souvent pour les équations trigonométriques, le programme l'effectue lorsque c'est possible.

Par ailleurs, lorsqu'on doit introduire les formules exprimant les lignes trigonométriques d'un arc en fonction de la tangente de l'arce moitié, on applique systématiquement le changement de variable inverse x ⟶ 2x de façon à ne pas avoir des $\frac{x}{2}$ partout dans l'expression transformée.

CONCLUSION

En conclusion de ce paragraphe sur la simplification des expressions, il nous semble important de signaler que la méthode a été conçue pour des équations à coefficients numériques. Nous avons pu cependant résoudre des équations comportant des paramètres formels à condition que ceux-ci ne nécessitent pas de traitements spécifiques.

III - REALISATION PRATIQUE

1) - Le programme :

Le programme a été écrit en FORTRAN CII et testé sur la CII 10070 de l'Institut Universitaire de Calcul Automatique de NANCY. Il comprend environ 1 600 instructions.
Les règles utilisées pour les simplifications, normalisations, calculs algébriques et trigonométriques sont lues et placées dans des tableaux dimensionnés à 3 000 mots. Les expressions générées dans l'arborescence de résolution sont dans un tableau PILEXP dimensionné à 4 000, ce qui permet 20 expressions contenant en moyenne une vingtaine de symboles.

Les temps d'exécution varient évidemment selon le nombre de symboles des expressions. Ils vont de moins de 10 secondes pour des équations simples résolues à l'aide des règles obligatoires à deux minutes pour les équations plus complexes.

2) - Liste d'équations résolues :

A partir d'une équation, notre programme trouve les équations simples équivalentes, nous n'avons pas programmé la résolution des équations algébriques obtenues, cela présentait un intérêt faible puisque ces résolutions sont purement algorithmiques.

Nous donnons ici la liste des équations résolues en indiquant pour chacune le nombre d'équations simples auxquelles le programme s'est ramené et le nombre de noeuds de l'arborescence générée. Nous indiquons également si les seules règles obligatoires ont permis d'aboutir à la solution.

Equation	Nombre d'équations engendrées	Solution obtenue par transformations obligatoires	Nombre total de noeuds des arborescences
$4 \sin^2 x - \frac{4}{\cos 2x} + 7 = 0$	1	oui	3
$\sqrt{3} \operatorname{tg} x - 4 \sin^2 x = 0$	2	oui	7
$\sin 4x + \sin 2x - 2 \sin 3x \cos 10x = 0$	3	non	8
$\cos^4 x + \cos^2 2x - \sin^4 x + 2 \sin^2 x - 1 = 0$	1	oui	2
$\operatorname{tg} x + \operatorname{tg} 3x = 0$	1	oui	3
$2 \cos^2 x + \sin 2x + \operatorname{tg} x = 0$	1	oui	3
$1 + \cos 2x + 2 \cos x = 0$	2	oui	4
$\sin^2 4x - \sin^2 x = 0$	4	oui	10
$5 \sin^2 x - \cos 2x - 3 = 0$	1	oui	2
$\operatorname{cotg}^2 x - \operatorname{tg}^2 x + 1 = 0$	1	non	3
$2 \cos 3x \cos x - 1 = 0$	1	non	3
$\sin x \cos x + \cos^2 x - \cos 2x - 1 = 0$	1	non	3
$\operatorname{tg} x - \frac{1}{\cos x} - \sqrt{2} = 0$	1	oui	3
$\cos x - \cos 2x - \sin 3x = 0$	2	non	10
$\sqrt{3} \cos 5x - \cos 2x - \cos 12x = 0$	2	non	5
$4 \cos^3 x \sin 3x + 4 \sin^3 x \cos 3x - 3 = 0$	1	non	5
$\cos 7x - \cos 4x + \cos x = 0$	2	non	6
$2 \sin^2 x + \sin^2 2x - 2 = 0$	2	oui	5
$\cos 2x - \cos 4x - \sin x = 0$	2	non	5
$\operatorname{tg} x + \operatorname{tg} 2x + \operatorname{tg} 3x + \operatorname{tg} 4x = 0$	3	non	15
$25 \sin^3 x + 20 \sin^2 x - 51 \sin x + 18 = 0$	1	immédiat	1
$\cos x + \sqrt{3} \sin x + \sqrt{2} = 0$	1	immédiat	1
$\operatorname{cotg} x - \operatorname{tg} x + 2 \operatorname{cotg} 4x = 0$	2	non	7
$\operatorname{tg} x + \operatorname{cotg} x - \operatorname{tg} 3x - \operatorname{cotg} 3x - 4 = 0$	4	non	13
$\sin 2x \operatorname{tg} 2x (4 - \operatorname{tg}^2 x) - m \operatorname{tg}^2 x = 0$	2	oui	6

Equation	Nombre d'équations engendrées	Solution obtenue par transformations obligatoires	Nombre total de noeuds des arborescences
$\cos 2x - \cos 2x \operatorname{tg} 2x \operatorname{tg} x - 1 = 0$	2	oui	6
$\sin 3x \sin x - 1 = 0$	1	non	3
$\sin 7x - \sin 3x - \sin x = 0$	2	non	5
$\sin x + 2 \cos x - \frac{1}{\cos x} = 0$	1	oui	3
$\sin x + \sin 2x + \sin 3x = 0$	2	non	6
$\cos x + \cos 2x + \cos 3x = 0$	2	non	6
$\sqrt{\sin x} + \sqrt{\cos x} - 1 = 0$	1	oui	6
$\sqrt{3 - 4 \cos^2 x} + 3 \sin x - 1 = 0$	1	oui	3
$\frac{\sin 3x}{\sin x} - \frac{\cos 3x}{\cos x} - 2 = 0$	1	non	3
$\operatorname{tg} x + \operatorname{tg} 2x - \operatorname{tg} 3x = 0$	2	non	12
$2 \sin^2 x + \cos^2 2x - 2 = 0$	1	oui	2
$\frac{3 \sin^2 x - 2 \sin x + 1}{2 + 3 \sin x} - \frac{1}{\sin x} = 0$	1	oui	2
$5 - 7 \sin x - \sqrt{1 - 4 \cos^2 x} = 0$	1	oui	3
$\cos 3x - \cos 2x = 0$	2	oui	3
$\sqrt{1 + \sin^2 x} + \sqrt{1 + \cos^2 x} - 1 = 0$	1	oui	4
$4 \sin 3x + \cos 2x - \sin x + 1 = 0$	1	non	3
$\frac{1}{\cos x + 1} + \frac{1}{\cos x - 1} - 1 = 0$	1	oui	2
$\frac{1}{\operatorname{tg} x} + \frac{1}{\operatorname{cotg} x} + 2 \sin 2x \sin x = 0$	1	non	7
$\sin 5x - \sin 3x + \sin x = 0$	2	non	5
$\sin 4x - \sin 2x + \cos 3x = 0$	2	non	5
$\sin 2x - \sin x = 0$	2	oui	4
$\cos 2x - \sin x = 0$	2	oui	5
$\sin 3x - \cos x = 0$	2	oui	5
$\sin^2 x - \frac{3}{\sin^2 x} = 0$	1	oui	2

Equation	Nombre d'équations engendrées	Solution obtenue par transformations obligatoires	Nombre total de noeuds des arborescences
$\frac{27}{\sin x} \operatorname{cotg} x \frac{8}{\cos x} \operatorname{tg} x = 0$	2	oui	6
$\cos^3 x - 2 \sin^2 x \cos x = 0$	2	oui	5
$\operatorname{tg} x - \cos x = 0$	1	oui	4
$\cos 2x - \cos x = 0$	2	oui	4
$\sin 2x \cos 2x + 2 \sin x = 0$	2	non	7
$\sin 4x - 2 \sin x \cos 2x = 0$	3	non	8
$\sin 2x - \cos x = 0$	2	oui	5
$\cos 5x - \cos 3x + \sin x = 0$	2	non	5
$\sin 4x + \sin 2x + 2 \cos x = 0$	2	non	5
$\cos 3x - \cos x = 0$	2	oui	4
$\operatorname{cotg} x - \cos x = 0$	2	oui	6
$\sin 4x - \sin x = 0$	2	oui	4
$\sin x + 2 \cos x \frac{1}{\cos x} = 0$	1	oui	3
$\cos 3x - \frac{2 \operatorname{tg} x}{1 + \operatorname{tg}^2 x} = 0$	2	non	10
$\operatorname{cotg} x + \operatorname{tg} x - \sin x - \cos x = 0$	1	oui	5
$\sin x \operatorname{tg} x + 2 \cos x - 1 = 0$	1	oui	4
$2 \sin^2 x - 3 \sin x \cos x + \cos^2 x = 0$	1	oui	2
$\operatorname{tg} x - 2 \operatorname{cotg} x + 3 = 0$	1	oui	3
$\sin^4 x + \sin^2 x - 2 = 0$	1	immédiat	1
$\operatorname{cotg}^2 2x - \operatorname{tg}^2 x + 1 = 0$	1	oui	3
$\sin x + \cos x - 3 \sin x \cos x - 1 + \frac{\sqrt{3}}{4} = 0$	1	oui	3
$\sqrt{2 \sin^2 x + 1} - 4 \sin x + 3 = 0$	1	oui	2
$\sqrt{1 + 2 \operatorname{tg}^2 x} - \operatorname{tg} x + 2 = 0$	1	oui	2
$\operatorname{tg} 2x + \operatorname{tg} x - \sin 3x = 0$	2	non	7
$\operatorname{tg} 3x - \sin 2x - \sin 4x = 0$	2	non	10

Equation	Nombre d'équations engendrées	Solution obtenue par transformations obligatoires	Nombre total de noeuds des arborescences
$\sqrt{2}\sin 2x - \cos 3x = 0$	2	non	7
$\sin x + \operatorname{cotg} x - \frac{1}{\sin x} - \cos x = 0$	2	oui	6
$\sin 2x + \operatorname{tg} x - \sin x = 0$	2	non	9
$\cos 3x - 3\cos 2x - 4\cos^3 x + \cos x - 1 = 0$	1	non	4
$2\sqrt{1 + \operatorname{tg} x}\operatorname{cotg} 2x + \operatorname{tg} x - 1 = 0$	1	oui	4
$\sin x + \cos x + \operatorname{tg} x + \operatorname{cotg} x + \frac{1}{\cos x} + \frac{1}{\sin x} = 0$	1	oui	5
$\sin\left(4x - \frac{\pi}{5}\right) - \cos\left(5x + \frac{\pi}{5}\right) = 0$	2	oui	3
$5\cos\left(x + \frac{\pi}{3}\right) + 4\cos\left(x - \frac{\pi}{5}\right) - h = 0$	1	non	2
$\sin\left(3x - \frac{\pi}{3}\right) - \cos\left(5x + \frac{\pi}{3}\right) = 0$	2	oui	3
$\frac{1}{\sin\left(\frac{\pi}{4} + x\right)} + \frac{1}{\sin\left(\frac{\pi}{4} - x\right)} - 2\sqrt{2} = 0$	1	non	5
$\sin 2x \operatorname{tg} 2x - m\frac{\operatorname{tg}^2 x}{4 - \operatorname{tg}^2 x} = 0$	2	oui	6
$\sin^6 x - \cos^6 x - m = 0$	1	oui	2
$\frac{\cos^2 a}{\cos^2 x} + \frac{\sin^2 a}{\sin^2 x} - 2m = 0$	1	oui	3
$\operatorname{tg} x - \operatorname{tg} 3x - m \operatorname{tg} 2x = 0$	3	non	17
$\sin 2x + \sin x - \cos x - m = 0$	1	non	5
$m\sin^4 x - \sin^2 x \cos^2 x + \cos^4 x - \cos 4x = 0$	1	non	3
$\operatorname{tg}\left(\frac{\pi}{4} + x\right) - m\operatorname{tg}^2 x \operatorname{cotg}\left(\frac{\pi}{4} - x\right) = 0$	1	non	3
$8\sin x \sin(a - x) - 3\cos^2 x = 0$	1	non	4
$\cos 3x - m\cos x \cos 2x = 0$	2	non	6

Equation	Nombre d'équations engendrées	Solution obtenue par transformations obligatoires	Nombre total de noeuds des arborescences
$\sin(n+1)x + \sin(n-1)x - \sin 2x = 0$	3	non	12
$\frac{2}{\cos x} - \frac{1}{\cos(x + 2x)} - \frac{1}{\cos(x - 2x)} = 0$	1	non	4
$(m + 1)\cos 4x - 8m(\cos^4 x - \sin^4 x) + 9m - 7 = 0$	1	non	3
$\sin 3x - a\sin^2 x = 0$	2	non	4
$\sin(a - x)\cos x - m\sin^2 x + 2 = 0$	1	non	4

3) - Quelques listings :

$$\text{Equation} \quad 4\sin^2 x - \frac{4}{\cos 2x} + 7 = 0$$

```
EQUATION
  +   *   4  ** SIN   X   2   + NEG   /   4 COS   *   2   X   7
REGLES OBLIGATOIRES
REGLE E
  + NEG   /   4 COS   *   2   X   + NEG   * COS   *   2   X   2   9
CH DE VARIABLE
  *   X   2
REGLE I
  + NEG   *  ** COS   X   2   +   * COS   X   9 NEG   4

SOLUTION IMMEDIATE
```

$$\text{Equation} \quad \frac{1}{\cos x + 1} + \frac{1}{\cos x - 1} - 1 = 0$$

```
EQUATION
  +   /   1   + COS   X   1   *   /   1   + COS   X NEG   1 NEG   1
REGLES OBLIGATOIRES
REGLE *
  + NEG  ** COS   X   2   *   * COS   X   2   1
SOLUTION IMMEDIATE
```


Equation $4\sin^2 x + \tan 2x - \tan 3x = 0$

```
EQUATION
  + TG   X   + TG   *   2   X NEG  TG   *   3   X
REGLES NON OBLIGATOIRES
RNO A
  +   / SIN   *   3   X   * COS   X COS   *   2   X NEG  TG   *   3   X
REGLES OBLIGATOIRES
REGLE I
  + NEG   *  TG   *   3   X   * COS   *   2   X COS   X SIN   *   3   X
C= 7
RNO A
  + NEG   / SIN   *   2   X   * COS   X COS   *   3   X TG   *   2   X
REGLES OBLIGATOIRES
REGLE I
  +   *  TG   *   2   X   * COS   *   3   X COS   X NEG SIN   *   2   X
C= 7
RNO A
  + NEG   / SIN   X   * COS   *   2   X COS   *   3   X TG   X
REGLES OBLIGATOIRES
REGLE I
  +   * COS   *   3   X   * COS   *   2   X TG   X NEG SIN   X
C= 7
RNO G
  + NEG   /   *   3  TG   X NEG  ** TG   X   3   + 1 NEG   *   3  ** TG
    X   2   + /   *   2  TG   X   + 1 NEG  **  TG   X   2  TG   X
REGLES OBLIGATOIRES
REGLE I
  +   * ** TG   X   5   2 NEG   *  ** TG   X   3   6
MIFAC
  *   + ** TG   X   2 NEG   3   * ** TG   X   3   2
NOUVELLE EQUATION
 ** TG      3
SOLUTION IMMEDIATE
NOUVELLE EQUATION
  + ** TG   X   2 NEG   3
SOLUTION IMMEDIATE
```

Equation $\sin 5x - \sin 3x + \sin x = 0$

```
EQUATION
  + SIN   *   5   X   + NEG SIN   *   3   X SIN   X
REGLES NON OBLIGATOIRES
RNO A
  +   * COS   *   4   X   * SIN   X   2 SIN   X
MIFAC
  *   +   *   2 COS   *   4   X   1 SIN   X
NOUVELLE EQUATION
SIN   X
SOLUTION IMMEDIATE
NOUVELLE EQUATION
  +   * COS   *   4   X   2   1
SOLUTION IMMEDIATE
```

Equation $\sqrt{3}\ \mathrm{tg}\ x - 4 \sin^2 x = 0$

```
EQUATION
  +   * RAC   3  TG   X NEG   *   4  ** SIN   X   2
REGLES OBLIGATOIRES
REGLE C
  +   /   * RAC   3 SIN   X COS   X NEG   *  ** SIN   X   2   4
MIFAC
  *   +   / RAC   3 COS   X NEG   *   * SIN   X SIN   X
NOUVELLE EQUATION
SIN   X
SOLUTION IMMEDIATE
NOUVELLE EQUATION
  +   / RAC   3 COS   X NEG   * SIN   X   4
REGLES OBLIGATOIRES
REGLE I
  + NEG   * SIN   X   * COS   X   4 RAC   3
REGLES OBLIGATOIRES
REGLE D
  +   *  **  TG   X   2 RAC   3   + NEG   *  TG   X   4 RAC   3
SOLUTION IMMEDIATE
```


Equation $4 \cos^3 x \sin 3x + 4 \sin^3 x \cos 3x - 3 = 0$

```
EQUATION
  +   *   4   *  ** COS   X   3 SIN   *   3   X   +   *   4   *  ** SIN   X   3
  COS   *   3 X NEG   3
REGLES NON OBLIGATOIRES
RNO F
  +   * SIN   *   3   X   * COS   X   3   +   * COS   *   3   X SIN   X NEG   1
  NEG   3
MIFAC
  +   * SIN   *   3   X COS   X   +   * COS   *   3   X SIN   X   3   * COS   X
C= 7
RNO G
  +   *  ** COS   X   3   * SIN   X  12   + NEG   *  ** SIN   X   3   * COS   X
  12 NEG   3
MIFAC
  +   *  ** COS   X   3   * SIN   X   4   + NEG   *  ** SIN   X   3   * COS   X
    4 NEG   1
C=13
REGLES NON OBLIGATOIRES
RNO C
  + SIN   *   4   X NEG   1
SOLUTION IMMEDIATE
```

Equation $2\sqrt{1 + tg\,x\,cotg\,2x} + tg\,x - 1 = 0$

```
EQUATION
  + * 2 RAC + 1 * TG X COT * 2 X + NEG 1 TG X
REGLES OBLIGATOIRES
REGLE A
  + * COT * 2 X * TG X 4 + NEG ** TG X 2 + * TG X
    2
    3
EXPRESSION SUPPOSEE POSITIVE
  + NEG TG X 1
REGLES OBLIGATOIRES
REGLE C
  + NEG * ** TG X 2 3 + * TG X 2 5
SOLUTION IMMEDIATE
```


4) - Cas d'échec - Améliorations souhaitables :

Le programme échoue sur une équation relativement simple :

$\sin 2x - \sin x - \cos 2x + \cos x - 1 = 0$

L'application des règles non obligatoires conduit à une équation du 4ème degré en $tg \frac{x}{2}$:

$$- 3\,t^4 - 6\,t^3 + 4\,t^2 + 2\,t - 2 = 0$$

alors qu'après application de la R.N.O. g on avait :

$$2 \sin x \cos x - \sin x - 2 \cos^2 x + \cos x = 0$$

équation sur laquelle il suffisait de faire deux mises en facteurs différentes sur deux parties de l'expression.

$$\sin x\,(2 \cos x - 1) - \cos x\,(2 \cos x - 1) = 0$$

soit

$$(2 \cos x - 1)\,(\sin x - \cos x) = 0$$

qui se décompose en deux équations dont la résolution est immédiate.

Il faudrait donc permettre au programme d'utiliser ce type de mise en facteurs lorsque les autres transformations prévues ont échoué. Cela demanderait de revoir la solution choisie pour la forme simplifiée des expressions, c'est en effet incompatible avec l'application systématique de la distributivité de * par rapport à +.

C'est le seul type d'échec que nous ayons rencontré.

Discussion d'une équation :

Nous sommes parfois amenés à remplacer une équation par une ou plusieurs autres dont l'ensemble des solutions n'est pas le même que l'ensemble des solutions de l'équation initiale.

Exemple : $\sqrt{3 - 4 \cos^2 x} = 1 - 3 \sin x$

Nous remplaçons cette équation par celle obtenue en élevant les deux membres au carré. Parmi les racines de la nouvelle équation, on ne devra conserver

que celles satisfaisant à la condition :

$$1 - 3 \sin x \geqslant 0$$

Exemple : $\dfrac{3 \sin^2 x - 2 \sin x + 1}{2 + 3 \sin x} - \dfrac{1}{\sin x} = 0$

Nous remplaçons cette équation par celle obtenue après réduction au même dénominateur :

$$3 \sin^3 x - 2 \sin^2 x - 2 \sin x - 2 = 0$$

Parmi les solutions de cette nouvelle équation, seules celles qui satisfont aux conditions suivantes doivent être retenues :

$$2 + 3 \sin x \neq 0$$
$$\sin x \neq 0$$

Ces exemples montrant qu'au cours de la discussion d'une équation, on est amené à résoudre d'autres équations ou inéquations, il suffirait donc de rappeler le programme pour obtenir les solutions nécessaires à la discussion.

CONCLUSION

Que souligner au terme de cette étude heuristique de résolution d'équations trigonométriques ?

Du point de vue de la méthode, le développement d'une arborescence de résolution avec des noeuds ET/OU, l'utilisation de transformations quasi algorithmiques quand c'est possible et une utilisation mesurée de toutes les autres règles sinon se sont ici encore révélées très efficaces.

Du point de vue de la manipulation des expressions, nous avons mis au point des procédures permettant de travailler sur des expressions assez complexes. D'autres programmes heuristiques dans des domaines voisins ont d'autres systèmes de calcul formel, il serait évidemment souhaitable d'harmoniser les représentations ; ces programmes pourraient alors travailler de façon complémentaire ou s'insérer dans un ensemble plus vaste.

Tel qu'il est, P.R.E.T. a fonctionné sur près d'une centaine d'exercices, de plus il contient en lui-même les éléments nécessaires pour mener une discussion selon un canevas donné. Le résultat est donc satisfaisant.

BIBLIOGRAPHIE

Ouvrages généraux concernant les Mathématiques et la Trigonométrie :

(1) CAMPBELL R.
La Trigonométrie.
(Coll. Que sais-je ?, P.U.F., PARIS 1956).

(2) CARONNET Th.
Exercices de Trigonométrie.
(Librairie Vuibert, PARIS 1934).

(3) COMMISSAIRE H., CAGNAC G.
Cours de Mathématiques.
(Classe de Mathématiques Supérieures, tome I, chap. IV)
(Masson, PARIS 1944).

(4) LESPINARD V., PERNET R.
Trigonométrie, classe de Mathématiques.
(Librairie A. Desvigne, LYON 1959).

Ouvrages et articles concernant le calcul formel :

(5) AMAREL S.
On representations of problems of reasoning about actions.
(Machine Intelligence 3, p. 131-171, 1968).

(6) BOBROW J.G. (Ed.)
Symbol manipulation and techniques.
Proceedings of the IFIP Working Conference on Symbol manipulation languages, PISA 1967.
(North Holland publishing Company, AMSTERDAM 1968).

(7) CAVINESS B.F.
On canonical forms and simplifications.
(J. ACM., vol. 17, n° 2, Apr. 1970, p. 385-396).

(8) COLLINS G.E.
A system for polynomial manipulation.
(C. ACM., vol. 9, n° 8, 1966, p. 578-589).

(9) MARTIN W.A.
Determining the equivalence of algebraic expressions by hash-coding.
(J. ACM., vol. 18, n° 4, Oct. 1971, p. 548-558).

(10) MOSES J.
Algebraic simplification : a guide for the perplexed.
(C. ACM., vol. 14, n° 8, Aug. 1971, p. 527-537).

(11) SIRET Y.
Contribution au calcul formel sur ordinateur.
(Thèse. GRENOBLE 1970).

Ouvrages et articles concernant l'Intelligence Artificielle et la Résolution de problèmes :

(12) BANERJI R.B.
Theory of Problem Solving : an approach to Articicial Intelligence.
(American Elsevier Publishing Company, NEW-YORK 1969).

(13) BLEDSOE W.W.
Splitting and reduction heuristics in automatic theorem-proving.
(A.I., vol. 2, n° 1, 1971).

(14) BLESDOE W.W., BOYER R.S., HENNEMANN W.H.
Computer proofs of limit theorems.
(A.I., vol. 3, n° 1, 1972).

(15) BRAFFORT P.
L'intelligence artificielle.
(P.U.F., PARIS 1968).

(16) CHANG C.L.
The Unit proof and the input proof in theorem-proving.
(J. ACM., vol. 17, n° 4, 1970, p. 698-707).

(17) CHANG C.L., SLAGLE J.R.
An admissible and optimal algorithm for searching AND/OR graphs.
(A.I., vol. 2, 1971, p. 117-128).

(18) CORAY G.
Intelligence Artificielle.
(Cours, 3ème Ecole d'Eté d'Informatique de l'A.F.C.E.T., 1973).

(19) DELHAYE J.L.
DATAL : Un programme de démonstration automatique des théorèmes en Algol.
(Thèse 3ème cycle, PARIS VI, 1970).

(20) ERNST G., NEWELL A.
GPS : A case study in generality and problem-solving.
(ACM. monograph series, Academic Press, NEW-YORK, 1969).

(21) FEIGENBAUM E.A., FELDMAN J. (Ed.)
Computers and Thoughts.
(Mc Graxx Hill, NEW YORK, 1963).

(22) FOX L. (Ed.)
Advances in programming and non numerical computation.
(Pergamon Press, 1966).

(23) GELERNTER H.
Realization of a geometry-theorem proving machine.
(in Computers and Thoughts, p. 134, Mac Graxx Hill, 1963).

(24) GILMORE P.C.
An examination of the geometry-theorem proving machine.
(A.I., vol. 1, n° 3, 1970).

(25) HOLDEN A.D.C., JOHNSON D.L.
The use of imbedded patterns and canonical forms in
a self-improving problem-solver.
(Proceedings of the ACM. National Meeting, 1967).

(26) JONCOUR Y.H.
Automatic Generation of proofs for trigonometric
identities.
(Thèse U.S.A., 1971).

(27) JOUANNAUD J.P., GUIHO G.
Intelligence artificielle et reconnaissance des formes.
(La Recherche, n° 43, mars 1974).

(28) LAURENT J.P.
Un programme qui calcule des limites en levant les indéter-
minations par des procédés heuristiques.
(Thèse 3ème cycle, PARIS VI, 1972).

(29) LAURENT J.P.
A program that computes limits using heuristics to
evaluate the indeterminate forms.
(A.I., vol. 4, n° 2, 1973).

(30) MELTZER B., MICHIE D. (Ed.)
Machine Intelligence (vol. 1, 2, 3, 4, 5, 6).
(American Elsevier Publishing Compagny, NEW YORK, 1967).

(31) MINSKY M.
Steps toward Artificial Intelligence.
(in Computers and Thoughts, p. 406, Mac Graw Hill, 1963).

(32) MOSES J.
Symbolic Integration ; the stormy decade.
(C. ACM., vol. 14, n° 8, Aug. 1971, p. 548-560).

(33) NILLSON N.J.
Problem solving methods in artificial intelligence.
(Mac Graw Hill, 1971).

(34) NEWELL A., SHAW J.C., SIMON H.A.
Empirical explorations with the logic theory machine :
a case study in heuristics.
(in Computers and Thoughts, p. 109, Mac Graw Hill 1963).

(35) NEWELL A., SIMON H.A.
GPS, a program that simulates human thought.
(in Computers and Thoughts, p. 279, Mac Grax Hill, 1963).

(36) PITRAT J.
Réalisations de programmes de démonstration de théorèmes
utilisant des méthodes heuristiques.
(Thèse, PARIS, 1966).

(37) PITRAT J.
Intelligence Artificielle et Méthodes heuristiques.
(Revue Française de Recherche Opérationnelle, n° 39,
Dunod, PARIS, 1966).

(38) PITRAT J.
Un programme de démonstration de théorèmes.
(Monographies d'Informatique A.F.C.E.T., Dunod,
PARIS, 1970).

(39) QUINLAN J.R., HUNT E.B.
A formal deductive problem-solving system.
(J. ACM., vol. 15, n° 4, Oct. 1968, p. 625-646).

(40) ROBINSON J.A.
A machine-oriented logic based on the resolution principle.
(J. ACM., vol. 12, n° 1, Jan. 1965, p. 23-41).

(41) SIKLÓSSY L., MARINOV V.
Heuristic search versus exhaustive search.
(2nd IJCAI, Advance Papers, 1971).

(42) SIKLÓSSY L., RICH A., MARINOV V.
Breadth-first search : some surprising results.
(A.I., vol. 4, n° 1, 1973).

(43) SLAGLE J.R.
A heuristic program that Solves Symbolic integration problems in freshman calculus, Symbolic Automatic INTegrator.
(Thèse, MIT Lincoln Laboratory, 1961).

(44) SLAGLE J.R.
A heuristic program that Solves Symbolic integration problems in freshman calculus.
(in Computers and Thoughts, p. 191, Mac Graw Hill, 1963).

(45) SLAGLE J.R.
Artificial Intelligence : the heuristic programming approach.
(Mac Graw Hill, 1971).

(46) VIVET M.
Un programme qui vérifie des identités en utilisant le raisonnement par récurrence.
(Thèse 3ème cycle, PARIS VI, 1973).

(47) WINOGRAD T.
Understanding natural language.
(Edinburgh University Press, 1972).

TABLE DES MATIERES