Reconnaissance de symboles techniques et analyse contextuelle de schémas

Anja Habacha Hamada

Institut National
Polytechnique de Lorraine

CRIN-CNRS
Inria Lorraine

Ecole doctorale IAE+M

Département de Formation
Doctorale Informatique

Reconnaissance de symboles techniques et analyse contextuelle de schémas

THÈSE

présentée le **24 juin 1993** pour l'obtention du

Doctorat de l'Institut National Polytechnique de Lorraine

(Spécialité Informatique)

par

Anja HABACHA HAMADA

Composition du jury :

Président : Roger MOHR

Rapporteurs : Jean-Paul HATON
Guy LORETTE
Heinrich NIEMANN

Examinateurs : Daniel COULON
Karl TOMBRE

Remerciements

Je suis heureuse aujourd'hui d'exprimer ma reconnaissance à :

Monsieur Roger Mohr, Professeur à l'Institut Polytechnique de Grenoble, d'avoir présidé ce jury,

Monsieur Jean Paul Haton, Professeur à l'universté de Nancy 1, Monsieur Guy Lorette, Professeur à l'univerité de Rennes et Monsieur Heinrich Niemann, Professeur de l'université de Erlangen-Nurnberg, pour avoir accepté d'être les rapporteurs de cette thèse. Qu'ils trouvent ici l'expression de ma gratitude pour l'intérêt qu'ils ont porté à mon travail de recherche,

Monsieur Daniel Coulon, Professeur à l'Ecole des Mines de Nancy pour avoir accepté d'examiner mon mémoire et pour l'intérêt qu'il a porté au travail,

Monsieur Karl Tombre, chargé de recherche INRIA-lorainne au Centre de Recherche en Informatique de Nancy et directeur de mes travaux, pour les conseils et les critiques constructives qu'il m'a prodigué.

Je tiens à remercier tous ceux qui m'ont conseillé et aidé à réaliser cette thèse dans de bonnes conditions et plus précisement , j'exprime mes remerciements chaleureux à Houda Chabbi et à Stéphane Paris de l'équipe MOVI du CRIN à qui je dois beaucoup pour les nombreuses et fructueuses discussions que nous avons eues ensemble au sujet de mon travail et pour avoir relu ma thèse avec autant de soin.

Je tiens également à exprimer mes remerciements à Brigitte Wrobel-Dautcourt pour avoir relu l'etat de l'art de ma thèse.

Enfin, merci de tout cœur à ma famille et à tous mes amis en particulier Julian Anigbogu, Narjes Doggaz et tous les autres pour leur sympathie et leur soutien amical.

Table des matières

1
Introduction générale

Le travail présenté dans ce mémoire s'inscrit dans le domaine de l'analyse de documents et plus particulièrement, l'analyse de documents techniques. Ces documents appelés "*schémas*" représentent des circuits électriques, électroniques ou réseaux téléphoniques. Un schéma comporte trois catégories de composantes essentielles : *le texte*, *les lignes de connexion* et *les symboles* (Figure 1.1). L'analyse de schémas revient alors à l'extraction de ces trois types de composantes et à leur reconnaissance.

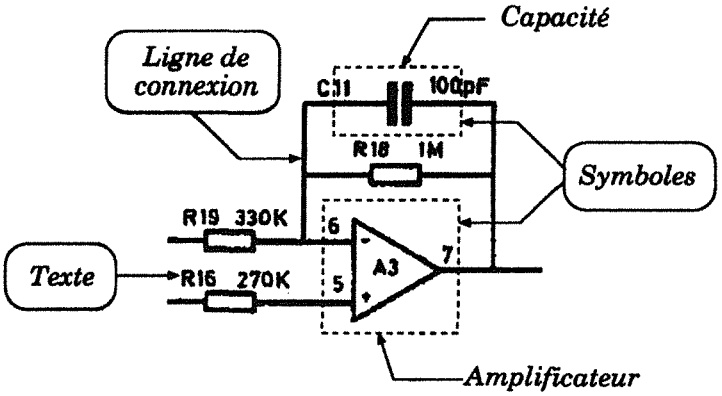


Figure 1.1. *Exemple d'un schéma : circuit électrique.*

Les schémas constituent, dans le monde industriel, un outil de communication important. L'analyse automatique de ces documents fournit un moyen de gestion efficace de l'information. En effet, l'informatisation permet le stockage des documents dans un minimum de place, mais aussi, permet leur réutilisation et leur transfert par réseaux ; ce qui facilite la communication. Ce besoin d'interprétation et de conversion automatique de documents en un format standard s'est accentué avec l'apparition des systèmes de Conception Assistée par Ordinateur (CAO) dans les entreprises. C'est pour toutes ces raisons que plusieurs systèmes d'analyse de documents techniques fondés sur des méthodes de reconnaissance de formes ont été réalisés.

Notre objectif est de développer une chaîne complète d'interprétation automatique de schémas. Pour le réaliser, nous proposons un système qui suit la méthodologie générale des systèmes de reconnaissance de formes et par conséquent, se décompose en deux niveaux comme le montre la Figure 1.2. Le *bas niveau* se charge d'extraire ce que l'on appelle les primitives images qui représentent les informations élémentaires de travail. Ces primitives contiennent des informations plus pertinentes que l'information pixel. Le *haut niveau* s'occupe de l'organisation et de la reconnaissance de ces primitives.

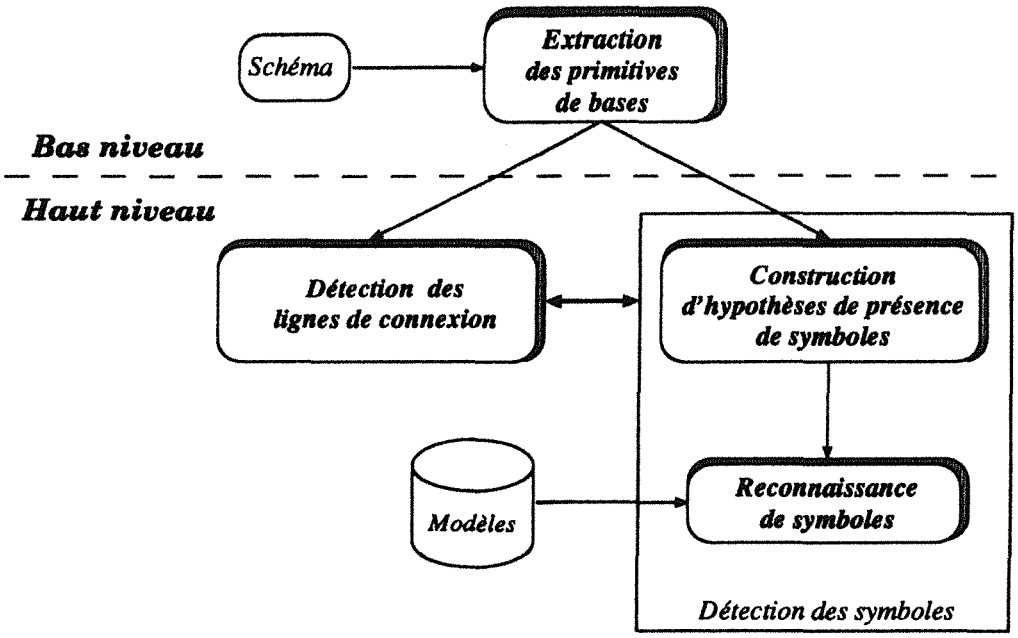


Figure 1.2. *Architecture générale de notre système.*

Dans notre cas, le bas niveau se charge d'extraire les primitives images qui sont les segments, les points de jonction et les composantes connexes et de localiser le texte[1]. Il utilise des outils du système REDRAW[2] [Antoine 90]. Quant au haut niveau, il a pour tâche de *détecter les lignes de connexion* et *les symboles* qui sont les deux composantes graphiques constituant un schéma.

Deux modules composent le haut niveau. Chacun de ces modules est réservé à la détection d'une des composantes. Il est à noter que ces deux modules ont une interaction mutuelle et forte due au *lien contextuel* qui existe entre les lignes de connexion et les symboles d'un schéma.

Le module de détection de symboles est fondé sur le principe de *prédiction-vérification d'hypothèses.* Les hypothèses représentent les zones du documents sus-

1. Notons que le problème de reconnaissance du texte n'est pas à l' dans ce mémoire.
2. REading DRAWings: développé dans notre équipe.

ceptibles de contenir un symbole et elles forment ainsi *les symboles candidats* à la reconnaissance. Ces hypothèses sont construites par l'étape de prédiction qui doit tenir compte du fait que les symboles peuvent être de deux types : d'une part ceux qui contiennent des boucles[3] comme une diode ou un amplificateur et d'autre part ceux qui ne contiennent pas de boucles comme par exemple une capacité ou encore une résistance (Figure 1.1). La délimitation de ces zones de présence de symboles utilise en plus de l'information *points extrémités* des lignes de connexion déjà détectées, l'information *boucle* et *texte* qui permet de prédire la présence de symboles ne contenant pas de boucles.

La *reconnaissance de symboles* supposés être contenu dans les zones de présence constitue l'étape de vérification des hypothèses. Ce processus de reconnaissance utilise une base de modèles connus a priori. L'approche adoptée pour la reconnaissance de symboles a le grand avantage d'être *invariante* par rapport à l'homothétie et la rotation qui existent entre le symbole et le modèle, et est fondée sur la *mise en correspondance structurelle* des graphes attribués.

Le module de détection des lignes de connexion est fondé sur l'utilisation du lien contextuel existant entre les symboles et les lignes de connexion. En effet, la détection des lignes de connexion est guidée par la reconnaissance de symboles puisque les lignes reliant le symbole reconnu au schéma sont des lignes de connexion. Cette utilisation des liens contextuels permet la détection des lignes de connexion sans aucune restriction ni sur la direction, ni sur la longueur de ces lignes. Par ailleurs, elle a l'avantage de réduire le nombre de fausses boucles utilisées. Une boucle est dite *fausse* si la chaîne contient des segments appartenant à des lignes de connexion.

Signalons enfin que notre système d'interprétation tient compte du bruit affectant les images qui se traduit par des informations manquantes ou mal extraites.

Plan de la thèse

Le plan de ce mémoire s'articule autour de six chapitres dont voici les centres d'intérêts :

- *le chapitre 2* comporte une présentation générale de l'état de l'art de l'analyse des documents et des problèmes de reconnaissance qui en découlent. Il se termine par une étude plus précise des travaux de recherche en analyse de schémas ;
- *le chapitre 3* présente les principes de notre système d'analyse contextuelle de schéma ainsi que les choix effectués pour réaliser nos objectifs. Nous décrivons ensuite les différents processus du bas niveau ;

3. Une boucle est une chaîne fermée de segments.

- *le chapitre 4* présente le processus de reconnaissance de symboles. Dans un premier temps, le problème de mise en correspondance d'un symbole candidat avec un symbole modèle est formulé. Trois contraintes d'appariement sont présentées pour permettre de tenir compte de la structure topologique et géométrique du modèle dans la phase de mise en correspondance. L'étape de vérification du résultat de la mise en correspondance qui utilise des informations contextuelles, est ensuite décrite. Enfin, nous validons le processus de mise en correspondance par des résultats expérimentaux et nous présentons les heuristiques adoptées pour le choix du meilleur modèle ;

- *le chapitre 5* décrit la structure complète du processus d'interprétation. Nous détaillons les processus de construction d'hypothèses de présence de symboles et de détection des lignes de connexion. Nous présentons ensuite le fonctionnement du processus complet d'interprétation. Ce chapitre se termine par une présentation de résultats expérimentaux ;

- *le chapitre 6* comporte une étude des perspectives envisagées où nous abordons le problème d'indexation de la base des modèles.

2
Etat de l'art

L'analyse des différents types de documents a fait l'objet de nombreux travaux de recherches. Nous les avons classés en deux groupes d'une part, *l'analyse de textes* et d'autre part *l'analyse de plans.* L'analyse de textes est la compréhension automatique de l'écriture tandis que l'analyse de plans est la compréhension automatique de documents techniques.

Nous commençons ce chapitre par une présentation des principes généraux d'un système d'analyse de documents. La section 2.1 expose les différentes étapes : saisie et segmentation d'un document, extraction des formes caractéristiques et reconnaissance de ces formes. Ensuite, la section 2.2 détaille diverses méthodes permettant d'effectuer la reconnaissance des formes. Enfin, la section 2.3 examine plusieurs systèmes d'analyse spécifiques aux schémas.

2.1 Introduction à l'analyse de documents

Traditionnellement, la forme principale de communication et de stockage de l'information était le *document papier.* Aujourd'hui cette tendance est entrain de s'estomper au profit de l'informatisation des documents qui est devenue indispensable. Ce besoin d'informatisation est encouragé par la facilité de la gestion de l'information par machine et également par la facilité de communication de ces informations via les réseaux connectant ces machines. L'informatisation des documents présente plusieurs avantages : comme *la réutilisation* des documents (modification, de copie), *le stockage* de l'information dans un minimum d'espace et *le transfert* par réseaux. Grâce aux techniques de traitements d'images et de reconnaissance des formes, le domaine de l'analyse de document réalise cet objectif. Ces documents sont de plusieurs sortes : composites, formulaires, postaux ou techniques ; et constitués de différents types d'information :

- *les photographies :* ces informations peuvent être des images binaires, à niveau de gris, tramées (dans les journaux) ou couleurs. En général, les techniques

d'analyse de documents ne s'intéressent pas à la reconnaissance des différents éléments de ces photographies mais cherchent plutôt à séparer les photographies des autres informations du document [Aubert 91] ;

- *le texte :* il existe actuellement des systèmes commerciaux de lecture optique (Optical Character Recognition OCR) utilisés pour la reconnaissance des caractères dans un document d'imprimerie [Anigbogu 92]. Bien que certains de ces systèmes aient atteint de bons taux de reconnaissance, il reste cependant beaucoup à faire en particulier dans la reconnaissance des images bruitées et de l'écriture manuscrite ;

- *le graphique :* composé de lignes de dessins, de régions pleines ou hachurées propres à décrire les documents techniques ; par exemple les plans cadastraux, les dessins industriels ou les schémas. Plusieurs travaux d'interprétation de ces documents sont menés en vue de leur conversion en format d'un logiciel CAO.

Par ailleurs, un processus d'analyse de documents se compose, en général, des étapes suivantes :

1. **Saisie des données et prétraitement :** les documents sont saisis par scanner et sont représentés par des images binaires dont la taille dépend de la résolution choisie. Comme ces images comportent du bruit (informations manquantes ou mal extraites), certains processus de prétraitement ont été développés en vue de réduire ce bruit.

2. **Segmentation :** après l'étape de saisie et de prétraitement, une étape de segmentation est réalisée afin de localiser et extraire séparément les photographies, le texte et le graphique.

3. **Extraction des formes caractéristiques :** ce sont les formes pertinentes existant dans les différentes parties d'un document comme par exemple les symboles d'un schéma.

4. **Reconnaissance de formes :** cette étape a pour but d'identifier les différentes formes caractéristiques extraites précédemment.

Les deux premières étapes constituent le processus *du bas niveau* tandis que les deux étapes suivantes se situent dans la partie *haut niveau* d'un système d'analyse de documents. Les étapes du bas niveau seront présentées dans le prochain chapitre. Nous allons maintenant nous intéresser aux problèmes attachés au haut niveau. L'étape d'extraction des formes étant spécifique au type de documents traités, la section suivante se focalise sur la reconnaissance des formes.

2.2 Techniques de reconnaissance des formes

La reconnaissance est l'étape fondamentale de tout système d'analyse de documents. Pour réaliser notre système, nous avons été confronté au problème du choix de la méthode de reconnaissance. Pour cela nous avons commencé par réaliser une étude bibliographique des méthodes généralement utilisées pour la reconnaissance afin de déterminer celles qui s'adaptent au mieux à notre problème. Les méthodes de reconnaissance de formes peuvent être classées en trois catégories : statistiques, fondées sur l'intelligence artificielle ou structurelles.

Les approches statistiques sont fondées sur les principes de *classification* : la reconnaissance revient à trouver la classe d'appartenance pour la forme recherchée. En revanche les approches fondées sur l'intelligence artificielle sont fondées sur le mécanisme *d'inférence*. Enfin les approches structurelles se partagent en deux groupes et diffèrent suivant le mécanisme de raisonnement qu'elles utilisent : *la mise en correspondance* et *l'analyse syntaxique*. Cependant l'idée sous jacente à ces deux types d'approches est l'utilisation *de l'information structurelle* pour la représentation des formes [Mohr 90]. Ces informations structurelles sont utilisées pour modéliser deux aspects importants :

- *la composition* d'une forme complexe en sous-formes,
- *les relations* qui peuvent exister entre ces sous-formes.

L'intérêt le plus évident des méthodes structurelles est de pouvoir traiter directement l'information structurelle qui n'apparaît pas dans les représentations vectorielles des méthodes statistiques.

Chacune de ces classes possède ses points forts et ses limites. Pour remédier à ces inconvénients, différentes méthodes ont été combinées pour faire naître les *approches hybrides*. Nous exposons ci après ces différentes méthodes de reconnaissance tout en citant quelques applications en analyse de documents ou en vision par ordinateur.

2.2.1 Approches statistiques

Ces méthodes sont utilisées pour classer les formes [Duda 73, partie I],[Kittler 87] [Niemann 90, chapitre 4],[Belaïd 92, chapitre 5]. Dans ce type d'approche, les formes sont représentées par des vecteurs de caractéristiques $\vec{x} = (x_1, \ldots, x_n)$ où les x_i sont les n mesures caractéristiques de cette forme. Le concept de classification des formes peut alors être vu comme un partitionnement de l'espace des caractéristiques de ces formes. Les classes des modèles sont aussi définies par des vecteurs caractéréstiques, la reconnaissance consiste à rechercher la classe qui minimise la distance entre le vecteur caractéristique de la forme et celui du modèle.

Ces approches tiennent compte du bruit et des déformations que peut subir une forme et elles sont, en général, moins coûteuses que les approches structurelles ou celles fondées sur l'intelligence artificielle. Mais ce type de méthode pose quelques problèmes : il n'est pas toujours évident de déterminer les bonnes caractéristiques qui représentent au mieux la forme ; elles sont souvent choisies de façon arbitraire. Le deuxième problème réside dans le fait que ce type d'approche ne fournit pas une description précise de la forme mais simplement une description de la classe à laquelle la forme appartient. Signalons également que l'approche statistique néglige complètement les relations qui peuvent exister entre les différentes caractéristiques. Cependant, elles ont été utilisées dans de nombreux travaux. Comme ceux de Bråten [Bråten 85] et Lin [Lin 85] pour la reconnaissance de symboles techniques. Ces techniques sont le plus souvent combinées avec d'autres approches par exemple avec l'approche syntaxique [Tsai 80, Duerr 79].

Les méthodes les plus connues sont les *bayésiennes*, les *stochastiques* et les *connexionnistes* dont voici une briève présentation.

A- Méthodes Bayésiennes

Etant donnés une forme x et un ensemble de classes modèles w_i, la reconnaissance consiste à rechercher la classe ayant la plus forte probabilité de contenir x.

Soient w_i, $(i = 1 \ldots N)$ les N classes modèles, $p(w_i)$ la probabilité de la classe w_i connue a priori et x une variable aléatoire représentant le vecteur caractéristique de la forme à identifier. Il s'agit de maximiser $p(w_i/x)$ pour $i \in [1 \ldots N]$ où $p(w_i/x)$ exprime la probabilité de classer x dans w_i. Cette probabilité est définie par le théorème de Bayes comme suit :

$$p(w_i/x) = \frac{p(x/w_i) * p(w_i)}{p(x)}$$

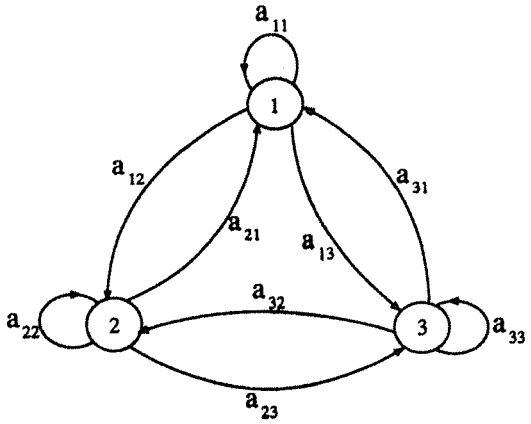
où $p(x/w_i)$ est la probabilité de x sachant qu'il appartient à w_i ; cette probabilité est estimée lors de l'apprentissage ; $p(x)$ désigne la probabilité de trouver une forme dont la représentation est x. Cette technique a été utilisée par Baird [Baird 91] pour la reconnaissance des caractères à multifontes fontes. Cet algorithme atteint un taux de reconnaissance de 99,7% des caractères sur des livres anglais selon la fonte.

B- Méthodes stochastiques

Dans ce type d'approche, la forme est considérée comme un signal continu observable dans le temps à différents endroits constituant des *états d'observation*. Le modèle décrit ces états à l'aide de probabilités de *transitions* d'état à état et de probabilités d'observation par état. La reconnaissance consiste à chercher dans ce

graphe d'états le chemin le plus probable correspondant à une suite d'éléments observés dans la chaîne d'entrée. Ces modèles sont robustes et fiables et la reconnaissance est rapide car les modèles comprennent généralement peu d'états.

Les modèles stochastiques les plus utilisés sont les *modèles de Markov cachés* HMM[1]. Dans ce cas, une forme est représentée par deux suites de variables aléatoires, l'une cachée et l'autre observable. La suite cachée S correspond à la suite d'états et la suite observable O correspond à la suite d'observations qui sont fonction du temps. La Figure 2.1 montre une modélisation possible d'une suite observée de nombres de 1 à 6 après une série de lancers de trois dés par un modèle de Markov caché . Il y a trois états dans le modèle correspondant chacun à un lancer d'un dé. Chaque état i est caractérisé par une distribution $d_i(j)$ de probabilités d'observer la valeur $j = 1, .., 6$ définissant ainsi la matrice $D = (d_{ij})$. Les transitions entre les états sont caractérisées par une matrice $A = (a_{ij})$.



O : 3 2 1 6 5 4 1 1 4 6 2
S : 2 1 1 3 2 1 3 1 3 2 2

		Observations					
		1	2	3	4	5	6
	1	$d_1(1)$	$d_1(2)$		...		$d_1(6)$
États	2	$d_2(1)$	$d_2(2)$		...		$d_2(6)$
	3	$d_3(1)$	$d_3(2)$		...		$d_3(6)$

Figure 2.1. *Modèle de Markov caché à trois états [Anigbogu 92].*

Étant donnés des modèles $\lambda_k = (A, D, \Pi)$ où A est la matrice de transition des états, D matrice de distribution des probabilités et Pi un ensemble de probabilités initiales de transition, et une suite d'observations O représentant une forme, le problème de reconnaissance consiste à déterminer le modèle λ_k tel que $P(O/\lambda_k)$ soit maximal où P est la probabilité que λ_k génère l'observation O. Dans notre laboratoire, cette technique est utilisée pour la reconnaissance de textes imprimés multifontes par Anigbogu [Anigbogu 92]. Le taux de reconnaissance varie de 96% à 99,6% selon la fonte.

1. Hidden Markov Model.

C- Réseaux neuronaux

Les réseaux neuronaux sont couramment proposés comme classifieurs statistiques. Plusieurs modèles existent, citons par exemple les cartes topologiques de Kohonen [Kohonen 89] et le modèle de Fukushima [Fukushima 88]. Un réseau est constitué de cellules ou *neurones formels* dont le fonctionnement s'inspire d'une modélisation des cellules neuronales biologiques. La Figure 2.2 illustre un neurone formel utilisé par [Fukushima 88]. Les $a(i)$ et b sont les poids des $N+1$ entrées réelles $u(i)$ et v. L'effet d'excitation du neurone est donné par $e = \sum_{i=1}^{N} a(i)u(i)$ et l'effet de l'inhibition du neurone est donné par $h = bv$. Le résultat fourni par la cellule est donné par la fonction :

$$\phi(x) = \begin{cases} x \ si \ x \leq 0 \\ 0 \ si \ non \end{cases} \quad avec \ x = \frac{1+e}{1-h} + 1$$

Ce qui traduit le fait que si $h > e$, le résultat fournit par la cellule est nul. Remarquons que dans d'autres types de neurones formels, le calcul de e peut être réalisé de façon *non linéaire* par rapport aux valeurs des entrées.

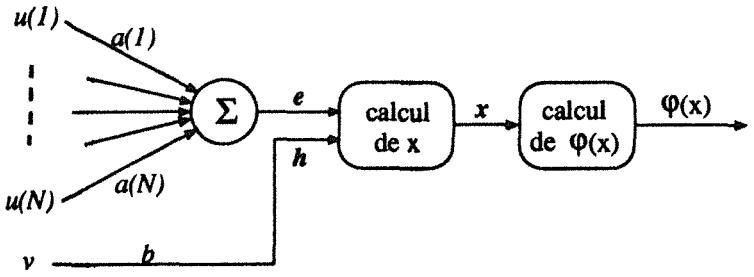
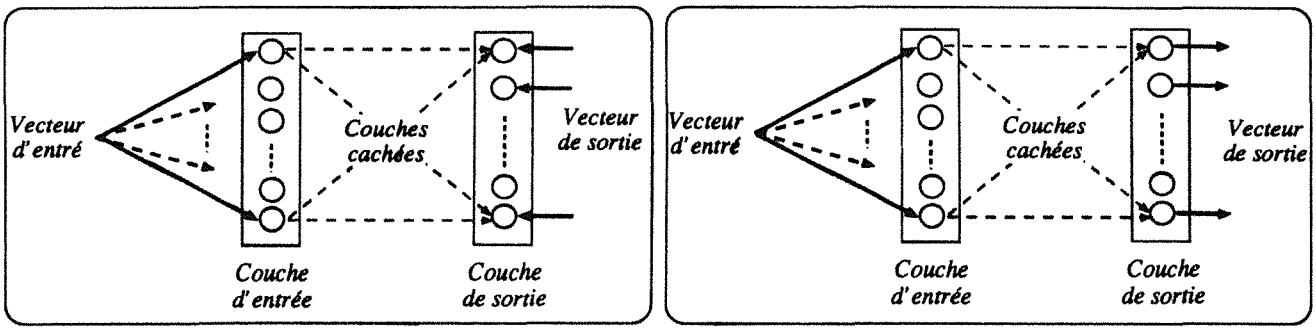


Figure 2.2. *Un exemple de neurone formel [Fukushima 88].*

Un système de reconnaissance de formes à base de réseau de neurones doit avoir deux modes de fonctionnement. La Figure 2.3 montre la configuration d'un réseau multi-couche dans chacun des deux modes :

- *mode d'apprentissage :* un algorithme d'apprentissage très utilisé est celui fondé sur la règle de *rétropropagation*. L'activation des cellules se propage jusqu'à la sortie du réseau, une erreur est alors calculée en comparant la sortie réelle et la sortie désirée. Cette erreur est rétropropagée et donne lieu à la modification des différents poids des entrées réelles des neurones. La phase d'apprentissage est réalisée jusqu'à convergence du système vers un taux de reconnaissance acceptable sur les formes d'apprentissage.

- *mode de reconnaissance :* le critère de discrimination permettant de juger la fiabilité de la reconnaissance est le maximum des erreurs commises.



a- Mode d'apprentissage ***b- Mode de reconnaissance***

Figure 2.3. *Configuration d'un réseau neuronal multi-couches.*

Ce type de modèle a été utilisé par Girod [Girod 92] pour la reconnaissance des caractères sur un support fortement bruité. Le problème consiste à reconnaître les dix chiffres (de 0 à 9) et les quatre lettres (de A à D) inscrits sur des billettes d'acier. Un réseau à deux couches de 14 neurones est utilisé. Un vecteur d'entrée donné est reconnu comme étant le caractère correspondant au neurone de sortie le plus actif.

2.2.2 Approches fondées sur l'intelligence artificielle

La caractéristique principale de ces approches est un mécanisme de raisonnement par "*inférence*". Les règles d'inférence utilisées dépendent du mode de représentation des connaissances qui n'est autre que celui de leur transcription sous forme symbolique pour qu'elles puissent être exploitées par un système de raisonnement. Plusieurs études [Haton 89][Niemann 90, chapitre 7] sur ces approches ont permis de formaliser le fonctionnement en deux objectifs non disjoints : représenter les connaissances et définir le mécanisme d'inférence permettant la résolution du problème. Plus précisément, nous expliquons par la suite les approches fondées sur la logique formelle, sur les règles de production, sur la représentation procédurale et sur les réseaux sémantiques.

A- Logique formelle

C'est une approche classique en intelligence artificielle pour la représentation et la déduction des connaissances. Plusieurs logiques sont connues, la plus répandue est le calcul des prédicats du premier ordre. Dans ce cas, les connaissances sont représentées par des formules bien formées suivant une syntaxe bien définie. Par exemple si $P(x,y)$ est un prédicat, qui signifie x possède y, la phrase "*un clavier possède des*

touches" est représentée par "$\exists x$ *clavier*(x) $\exists y$ *touche*(y) $P(x,y)$". Les règles d'inférence réalisant le raisonnement dans ce cas, sont *le modus ponens, le modus tollens, la spécialisation universelle et la résolution* [Haton 89][Niemann 90, chapitre 7]. Ces règles permettent la preuve de théorèmes ou la génération de nouvelles formules bien formées.

L'avantage du calcul des prédicats du premier ordre est qu'il est fondé sur un formalisme rigoureux et sur des bases théoriques solides. Il est bien adapté au raisonnement exact. Mais cette rigueur pose un problème: le fait que les prédicats ne prennent que les valeurs "vrai" ou "faux" implique que l'incertain ou le flou ne peuvent pas être formalisés. Pour remédier à ce problème, d'autres logiques non standard ont été étudiées, par exemple les logiques multivaluées, les logiques modales, les logiques non monotones [Sabah 88, Haton 89].

B- Systèmes de production

Un système de production est principalement constitué de trois composantes: une base de faits, une base de connaissances et un interprète ou moteur d'inférences [Laurière 86]. La base de connaissances est un ensemble de règles où chacune de ces règles représente une partie de la connaissance du problème. Une règle est généralement de la forme "si *prémisse ou condition* alors *action ou conclusion*". Chaque fois que la partie condition correspond à une situation dans la base de faits, la règle est dite activable. Les conclusions en découlent. La base de faits contient les conclusions spécifiques, produits par l'état actuel des données. L'ordre dans lequel les règles sont appliquées est un problème très important et caractérise le moteur d'inférence.

Deux mécanismes de raisonnement sont appliqués dans ce cas: le chaînage-avant qui est un raisonnement guidé par les données et le chaînage-arrière qui est guidé par l'objectif à atteindre. Dans le chaînage-avant, le moteur d'inférences sélectionne une règle parmi celles activables et l'applique en exécutant la partie conclusion de la règle. Dans le chaînage-arrière, l'interprète met en correspondance le but à atteindre avec une partie conclusion de l'une des règles de production du système, les prémisses de cette règle deviennent le nouveau but à atteindre.

En résumé, le moteur d'inférence fonctionne en trois étapes successives:

1. la recherche des règles appliquables,

2. la résolution des éventuels conflits, il faut choisir une règle si plusieurs règles s'appliquent,

3. l'activation de la règle choisie et la modification du contexte en conséquence.

Citons par exemple les travaux de Bley [Bley 84] qui utilise les règles de production pour résoudre les cas de conflits dans la vectorisation des circuits électriques.

Les systèmes de production possèdent de nombreux avantages. Les connaissances sont exprimées de manière déclarative. Le système est totalement modulaire du fait que les connaissances sont formalisées par des règles indépendantes les unes des autres. Par conséquent, il est facile d'ajouter, de supprimer ou de modifier des règles mais bien sûr il faut contrôler le maintient de la cohérence de l'ensemble des règles. En revanche, les systèmes de production deviennent rapidement inefficaces dès que le nombre de règles devient trop important car, dans ce cas, le système est obligé d'explorer toute la base de règles à chaque cycle et en est donc ralenti. Pour éviter cela, il faut structurer la base de règles en leurs attribuant un ordre de priorité, ce qui n'est pas toujours évident. Ce mode de représentation est quelque fois insuffisant puisque le coté procédural de la connaissance est complètement négligé.

C- Systèmes procéduraux

Dans les systèmes déclaratifs, la vision du monde est tout à fait statique: les connaissances sont données telles quelles, sans aucune information particulière à propos de leur utilisation. Par contre dans les systèmes procéduraux [Laurière 86], les connaissances sont contenues dans des programmes et des sous-programmes attachés à des tâches bien spécifiques. Notons que la différence essentielle entre ces deux approches réside dans le mécanisme de raisonnement. En effet, en déclaratif, l'information est claire, facilement accessible et la façon dont elle est utilisée est implicite en revanche, en procédural, le contrôle est clairement exprimé et la connaissance est plus ou moins dissoute dans les procédures et les représentations manipulées. Ceci implique que les systèmes sont figés et la modularité des systèmes de production est perdue. En général, l'attachement procédural est souvent inclu dans d'autres modes de représentation de connaissances par exemple les objets structurés.

L'une des applications dans notre équipe de ce mode de représentation est celle de l'analyse des plans cadastraux français [Antoine 91].

D- Réseaux sémantiques

Un réseau sémantique est un graphe où les nœuds représentent des concepts et les arcs des relations de nature sémantique entre les concepts. Les liens peuvent traduire des relations entre concepts génériques (≪sorte_de≫), entre ces concepts et des individus particuliers (≪est_un≫) ou d'autres relations qui traduisent une action. Les résaux sématiques sont bien adaptés à la représentation des concepts hiérarchisés. Les liens *sorte_de* et *est_un* permettent d'utiliser le mécanisme *d'héritage* de propriétés afin d'éviter la redondance d'information. L'un des principaux

mécanismes de raisonnement dans ce formalisme est "l'héritage de propriétés". Ce mécanisme assure la transmission des informations décrites dans un objet d'une classe supérieure aux objets des classes inférieures. Mais il n'est pas facile d'introduire et de manipuler dans les réseaux sémantiques une connaissance procédurale. L'utilisation des objets structurés ou frames, comme nœuds du réseaux, peut être considérée comme une solution à ce problème.

Une application des réseaux sémantiques dans le domaine de la vision permet la localisation des réservoirs d'aération dans une image aérienne d'un réseau de traitement d'eaux [Ballard 82, chapitre 10].

2.2.3 Approches syntaxiques

Les méthodes syntaxiques sont fondées sur la théorie des langages et des automates. Les relations structurelles dans les modèles sont définies de façon implicite et les formes sont représentées par le langage engendré par une grammaire formelle. Plusieurs chercheurs se sont intéressés aux approches syntaxiques [Fu 74, Sanfeliu 86]. Les symboles terminaux de la grammaire correspondent aux primitives, formes de base indécomposables, extraites directement de l'image par un processus de bas niveau. Les symboles non-terminaux décrivent les compositions autorisées des sous-formes. Le processus de construction de ces formes complexes est décrit par une grammaire et le mécanisme de reconnaissance revient à *une analyse syntaxique* de l'image donnée en utilisant cette grammaire. Notons que les formes décrites par la répétition d'une même sous-forme sont correctement décrites grâce au caractère récursif des grammaires. Par exemple, la règle $X = AB/AX$ d'une grammaire représente la description d'une forme X suivant une suite de sous-formes A se terminant par une sous-forme B. En revanche, suivant le type de la grammaire, l'analyseur syntaxique est plus ou moins complexe. Cependant, le bruit et les déformations, inhérents aux images, limitent l'utilisation de ces grammaires qui tolèrent difficilement l'incertitude. Plusieurs chercheurs ont étudié ce problème [You 80, Sanfeliu 86, Tsai 90].

Les grammaires formelles ne sont pas suffisantes et assez peu employées en reconnaissance de formes puisqu'elles ne permettent pas de décrire des relations n-aires entre les sous formes ; seule la concaténation à gauche et à droite est permise. Pour ces raisons, plusieurs autres types de grammaires ont été définies et utilisées. Citons par exemple les approches syntaxiques, utilisables pour différents objectifs, puisqu'elles fournissent non seulement une classification des formes mais aussi une description structurelle de la forme inconnue.

A- Les grammaires attribuées

Les grammaires attribuées permettent d'introduire des informations sémantiques et d'associer à chacune des règles un ensemble d'attributs numériques. Parmi les avantages de ces grammaires nous pouvons citer : l'amélioration des taux de reconnaissance des formes, la capacité de représenter une partie des informations sématiques propres aux formes.

Une grammaire attribuée a été utilisée dans les travaux de Bunke [Bunke 82] pour la reconnaissance de symboles et combinée à une approche statistique dans les travaux de Tai [Tai 90] pour la reconnaissance des caractères chinois.

B- Les plex-grammaires

Les plex-grammaires permetttent de décrire les relations n-aires et par conséquent, la juxtaposition des formes est effectuée sur plusieurs *points priviligiés* du contour des formes. Elle a été utilisée dans notre équipe par S. Collin [Collin 92] pour la reconnaissance de la cotation dans un dessin technique. La Figure 2.4 montre un exemple de superposition des points priviligiés de primitives formant une cotation, définie par un ensemble de sous-formes connectées par un ou plusieurs points particuliers : une sous-forme est un ensemble de primitives graphiques : *segments*, *flèches* et *bloc de texte* connectés par l'intermédiaire de certains points appelés *pôles*.

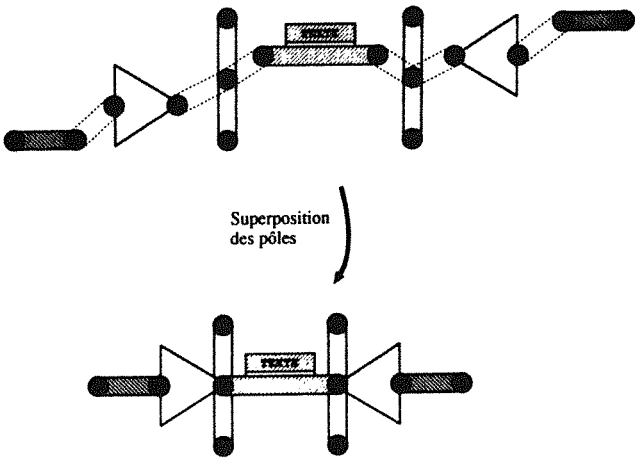


Figure 2.4. *Superposition des pôles des primitives graphiques [Collin 92].*

C- Les grammaires de graphes

Dans ce cas, les formes sont représentées par des graphes. Pour une règle de la grammaire de graphe $A \longrightarrow B$, il faut préciser comment remplacer le sous-graphe A d'un graphe G par le sous-graphe B. Donc, dans ce type de grammaires, il est

nécessaire de disposer pour chaque règle, d'une fonction qui précise le remplacement de A par B indépendemment du graphe G. Citons par exemple les travaux de Fahmy [Fahmy 92] où une grammaire de graphe pour la reconnaissance des notes musicales est utilisée. La Figure 2.5 montre un exemple de règles d'une grammaire de graphes où la condition d'application porte sur la distance entre la *clé* et la *note*, la clé doit être à gauche de la note.

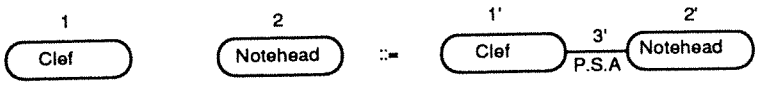


Figure 2.5. *Exemple de règle de grammaires de graphes [Fahmy 92].*

Généralement les grammaires de graphes sont utilisées en associant un ensemble d'attributs numériques à chacune des règles ; ceci donne naissance aux *grammaires de graphes attribuées* [Bunke 82]. Ceci permet de profiter des avantages des grammaires de graphe ainsi que de ceux des grammaires attribuées. Mais la faiblesse des grammaires de graphes réside dans la difficulté de mettre en œuvre un analyseur efficace [Bunke 82].

2.2.4 Mise en correspondance structurelle

La mise en correspondance est tout aussi utilisée dans le domaine de reconnaissance de formes qu'en vision par ordinateur. Ce type de méthode est orienté vers une description *géométrique* des formes. Contrairement aux techniques précédentes, les relations structurelles dans les modèles sont définies de façon explicite. L'idée consiste à sauvegarder explicitement un *nombre fini* de modèles qui ne sont plus décrits par des langages formels comme pour l'approche syntaxique, ni par des vecteurs de caractéristiques comme dans les approches statistiques mais par *des chaînes*, *des arbres* ou *des graphes.*

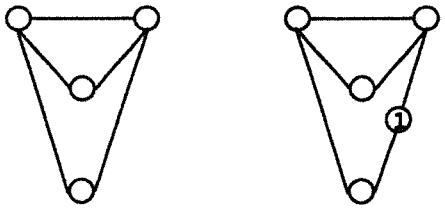
La procédure de reconnaissance associée est fondée sur la mise en correspondance avec un modèle par une coïncidence directe ou une mesure de similarité. Pour reconnaître une forme inconnue x, il faut mettre en correspondance x avec tous les modèles dans le but de détecter quel modèle est le plus similaire avec x.

Le mode de représentation des formes a une influence fondamentale sur le mécanisme de mise en correspondance et précisément sur la complexité en temps d'exécution. Pour la représentation en chaînes ou en arbres, le problème d'appariement est résolu en un coût polynomial [Bunke 90b, Sanfeliu 90]. En revanche, la mise en correspondance des graphes reste un problème difficile car la complexité en temps

d'exécution est exponentielle, cependant cette représentation est la plus importante et la plus utilisée.

Nous nous intéressons au problème de mise en correspondance de graphes qui s'identifie au problème de recherche *d'homomorphisme ou d'isomorphisme* entre deux graphes [Shapiro 81] [Ballard 82, chapitre 11]. En général, ce problème se ramène à un problème de recherche d'isomorphisme de *sous-graphes* [Ullmann 76] défini de manière suivante :

Définition 1 *Etant donnés deux graphes* $(\mathcal{N}_1, \mathcal{A}_1)$ *et* $(\mathcal{N}_2, \mathcal{A}_2)$, *il existe un sous-graphe de G isomorphe à H s'il existe un sous-ensemble* $N \subseteq \mathcal{N}_1$ *et un sous-ensemble* $A \subseteq \mathcal{A}_1$ *tel que* $|N| = |\mathcal{N}_2|$, $|A| = |\mathcal{A}_2|$ *et s'il existe une fonction un à un* $f : \mathcal{N}_2 \longrightarrow N$ *tel que* $\{u, v\} \in \mathcal{A}_2$ *si est seulement si* $\{f(u), f(v)\} \in E$.



(a) *(b)*

Figure 2.6. *Nécessité de la tolérance de la définition d'isomorphisme de graphes.*

Dans la pratique, la mise en correspondance ne peut pas respecter ces règles strictes de recherche d'isomorphismes à cause de la présence du bruit. En effet, la Figure 2.6 montre qu'il n'existe pas un sous graphe de (*b*) isomorphe au graphe (*a*) en vérifiant la définition 1. Mais en supposant que le nœud **1** soit de trop, les deux graphes sont alors isomorphes.

Afin de prendre en compte le bruit, des algorithmes de mise en correspondance *inexacte* ont été développés [Shapiro 80] : aux définitions classiques d'homomorphismes (resp d'isomorphismes) de deux graphes, Shapiro et Haralick [Shapiro 81] ont ajouté la notion de ϵ-homomorphismes (resp ϵ-isomorphismes), ϵ étant le seuil limite que peut atteindre le cumul des pondérations des relations non vérifiées. De cette façon, les relations entre les nœuds du graphe n'ont pas le même poids. Ceci a pour intérêt de tolérer que certains arcs du premier graphe peuvent ne pas avoir de correspondants dans le deuxième graphe. Puisque ce problème d'isomorphisme de sous-graphes est NP-difficile, l'utilisation des heuristiques est inévitable surtout si le graphe est de taille importante.

Ci après, nous présentons différentes techniques de mise en correspondance pour les représentations fondées sur les graphes.

A- Recherche arborescente

Cette stratégie consiste à considérer l'arbre des correspondances possibles. Dans ce cas, la solution du problème de mise en correspondance revient à chercher un *chemin* optimal dans cet arbre. Les méthodes de recherche d'un chemin dans arbre peuvent être classées en deux catégories :

- *recherche en profondeur d'abord*: chaque branche est entièrement explorée avant de passer à la branche suivante,
- *recherche en largeur d'abord*: les branches sont explorées parallèlement.

Le problème essentiel dans les méthodes de recherche de chemin est de réduire l'espace de recherche afin d'éviter une recherche exhaustive et par conséquent, d'utiliser des heuristiques pour élaguer très rapidement certaines branches de l'arbre. Par exemple, dans la méthode *forward-checking* [Shapiro 80, Haralick 80], l'exploration est en profondeur d'abord et tient compte des compatibilités des hypothèses futures des nœuds non encore traités avec les hypothèses contenues dans la branche traitée. Dans la méthode *looking-ahead* [Shapiro 80, Haralick 80], l'algorithme de recherche de chemin tient compte également de la compatibilité des hypothèses futures des nœuds non encore traités avec celles des nœuds non encore traités. Deux applications du premier algorithme sont réalisées dans [Granger 85] et effectuent la mise en correspondance d'une part de deux vues aériennes et d'autre part, d'images d'empreintes digitales.

B- Prédiction-Vérification

Cette approche a été utilisée par beaucoup de systèmes [Lux 85, Ayache 86, Ayache 87, Quan 89] car elle est bien adaptée à la vision artificielle. Elle est aussi utilisée en analyse de texte [Ouladj 89] dans la phase de reconnaissance de mots. Cette technique correspond au schéma classique de recherche avec retour arrière.

La prédiction consiste à engendrer de nouvelles hypothèses d'appariement en exploitant les contraintes impliquées par les hypothèses existantes tandis que la vérification consiste à voir si l'ensemble des hypothèses ainsi formé est cohérent. Si une incohérence apparaît, un retour arrière est effectué jusqu'au dernier état cohérent déjà créé. La solution finale est l'ensemble des hypothèses *cohérentes*.

La prédiction est un moyen de réduire l'explosion combinatoire, mais dans le pire des cas, cet algorithme de prédiction-vérification est exponentiel. La rapidité de la détermination de l'ensemble des hypothèses cohérentes dépend de la manière avec laquelle se fera la prédiction. Mohr [Mohr 88a] propose des règles générales pour la génération des hypothèses :

- l'hypothèse initiale d'appariement doit être suffisamment robuste pour ne pas être fausse,
- la génération des hypothèses suivantes doit être guidée par les choix précédents. Par exemple, dans le problème de localisation d'objet 2D dans une image [Ayache 86], la détermination de quelques hypothèses initiales permet d'estimer la transformation qui existe entre le modèle et l'objet et par conséquent, la prédiction de la position des primitives encore non identifiées devient immédiate. La combinatoire est donc importante au début de l'appariement, mais très rapidement l'estimation de la transformation va conduire à une suite quasi-déterministe d'appariements.

La détermination de l'hypothèse initiale reste un problème important. Elle dépend essentiellement de l'application et il n'est pas toujours très évident de pouvoir déterminer une correspondance qui soit sûre. Par exemple dans [Groen 85, Lee 90], la reconnaissance de symboles dans un schéma est fondée sur la technique de prédiction vérification et le choix de l'hypothèse initiale est réalisée en appariant quatre points du modèle sélectionnés à la main lors d'une étape d'apprentissage.

C- Transformée de Hough

La transformée de Hough [Duda 72, Muller-Belaïd 84, Illingworth 88] est une méthode générale permettant de détecter ou de rechercher certaines formes *paramétrées* ou *fixes définies modulo une transformation géométrique* [Mohr 88a]. Un exemple typique d'une forme paramétrée est une droite d'équation $y = ax + b$.

Cette méthode consiste à passer d'un ensemble d'indices images à l'espace des paramètres associés. Cette application est souvent surjective, à un indice correspond un ensemble de points dans l'espace des paramètres. Nous cherchons ensuite les points d'accumulation parmi l'espace des paramètres. Une forme constituée d'indices ou une propriété de forme est définie dans l'espace des paramètres par un point d'accumulation. La Figure 2.7 illustre ce principe pour la recherche de la droite d'équations $y = a'x + b$: chaque point (x,y) dans (a) correspond à une droite d'équation $b = -xa + y$ dans (b). Autrement dit, tous les points vérifiant l'équation $y = ax' + b'$, décrivent des droites, d'équation $b = -xa + y$ dans l'espace des paramètres, qui s'intersectent en un même lieu. En théorie ce lieu est le point (a', b') mais compte tenu du bruit dans les images, ce lieu est un point d'accumulation, c'est-à-dire une petite zone. Le problème de recherche des formes dans l'espace des indices images est ainsi transformé en un problème de recherche de points d'accumulation dans l'espace des paramètres.

La transformée de Hough est une méthode globale qui résiste bien au bruit dans l'image. Néanmoins elle est limité par le fait que seules les formes paramé-

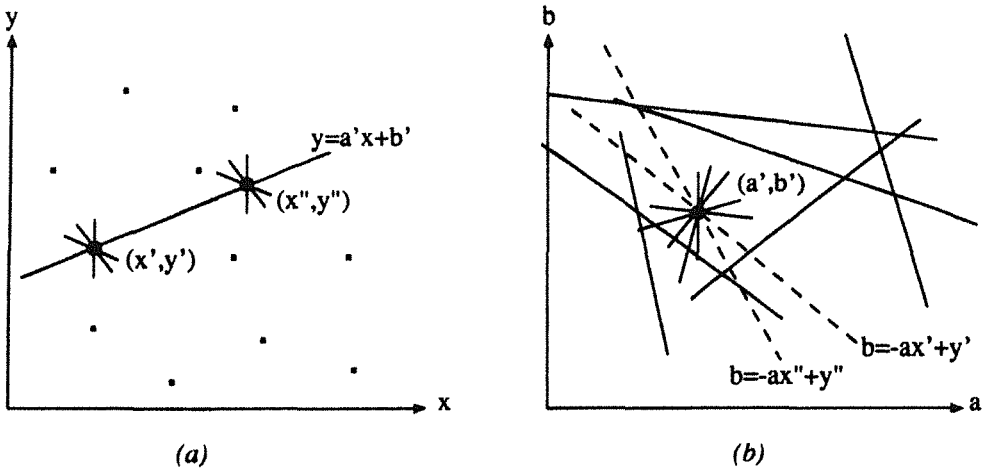


Figure 2.7. *Application de la transformée de Hough dans le cas de la recherche de droite dans une image [Ballard 82].*

triques sont identifiables. Dans notre équipe, elle a été utilisée par [Muller-Belaïd 84] pour la recherche de plans et de quadriques dans des images tridimentionnelles et par [Quan 89] pour la recherche des points de fuites dans une image.

D- Recherche de la Clique maximale

Cette approche consiste à construire un ensemble maximal d'hypothès mutuellement compatibles. Elle nécessite la construction préalable du graphe de compatibilité d'hypothèses. Les nœuds de ce graphe représentent les couples appariés (indice modèle, indice image) et les arcs représentent la relation de compatibilité. Le meilleur appariement correspond à la clique maximale du graphe de compatibilité entre hypothèses où une clique est un sous-graphe complet. La Figure 2.8 montre un exemple de graphes de compatibilité d'hypothèses. Chaque nœud correspond à une hypothèse d'appariement. Le groupe (1,a),(2,b),(3,c),(4,d) est une clique maximale; elle ne peut être étendue. Le groupe (1,a),(5,b) est aussi une clique de taille maximale, mais bien petite.

Cette technique s'exécute en deux étapes: la *construction du graphe de compatibilité* et la *recherche de la plus grande clique maximale* de ce graphe. La première étape nécessite qu'on soit capable de calculer pour tout couple d'hypothèses la valeur de la relation de compatibilité. En d'autres termes, savoir si tout couple du graphe de compatibilité est consistant ou non. Pour remédier à cette contrainte, Skordas et Horaud [Horaud 89] ont utilisé la fermeture transitive de la relation de compatibilité. Cette solution peut engendrer des incompatibilités et donc il ne faut pas abuser

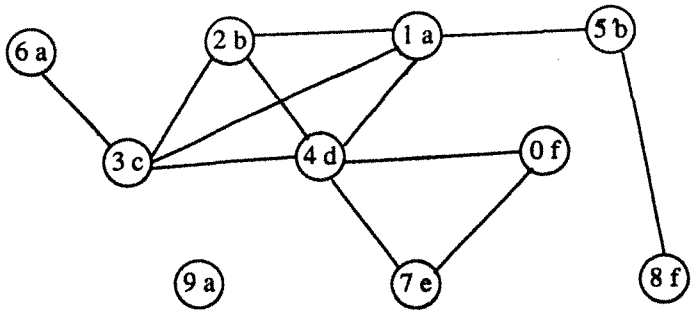


Figure 2.8. *Exemple de cliques maximales [Mohr 88a].*

de ce genre d'opération. Le deuxième problème de recherche de la plus grande clique maximale est un problème exponentiel puisque le seul algorithme qui existe effectue une recherche exhaustive de toutes les cliques maximales puis sélectionne la plus grande. La seule condition qui permet de réduire cette complexité est d'avoir un graphe de compatibilité de petite taille (ne dépassant pas 90 nœuds). Mais ceci n'est pas toujours possible. Cette technique a été appliquée pour la reconnaissance d'objets plats pouvant se recouvrir [Bolles 82].

E- Les techniques de relaxation

La relaxation est une technique opposée de la clique maximale. Au lieu de déterminer un ensemble maximal d'hypothèses mutuellement compatibles, l'idée est d'éliminer des hypothèses incohérentes en se fondant sur une compatibilité locale. Elle se décompose en deux techniques : la *relaxation discrète* et la *relaxation continue*. La première supprime les hypothèses incompatibles alors que la deuxième ajuste leur score.

Généralement, la convergence vers une hypothèse unique pour chaque indice image n'est pas assurée. Par conséquent, il vaut mieux la considérer comme un prétraitement permettant de restreindre l'espace des hypothèses.

• **Relaxation discrète :** Cette technique opère en deux étapes. La première affecte à chaque nœud (indice modèle) du graphe un ensemble d'étiquettes (indices images) qui peuvent lui correspondre. Ces étiquettes sont liées par des contraintes qui créent un réseau d'arcs sur ces nœuds, le transformant en un graphe. Ensuite un élagage de toutes les étiquettes incohérentes avec les étiquettes des nœuds voisins est réalisé.

Mohr et Henderson ont proposé [Mohr 86] [Henderson 90] un algorithme de consistance d'arcs, baptisé *AC4*, optimal en temps d'exécution et avec la même

complexité en espace mémoire que ceux présentés dans [Mackworth 85]. Mais il ne fournit que la consistance locale du graphe étiqueté et par conséquent, l'utilisation de cet algorithme nécessite une étape supplémentaire de vérification de la consistance globale du graphe. La Figure 2.9 montre que la consistance d'arcs n'est pas équivalente à la consistance globale puisque ce problème n'admet pas de solution malgré les appariements localement consistants. En effet, si on regarde les nœuds du graphes deux à deux et si le premier nœud est étiqueté par a alors le deuxième nœud admet l'étiqutte b et réciproquement. Par conséquent, la consistance locale est vérifiée bien que, globalement, l'étiquetage de trois nœuds par trois étiquettes toutes différentes est impossible. Une généralisation $GAC4$ de cet algorithme aux contraintes n-aire est présentée dans [Mohr 87, Mohr 88b], et nous trouvons une implantation de $GAC4$ dans [Paris 88].

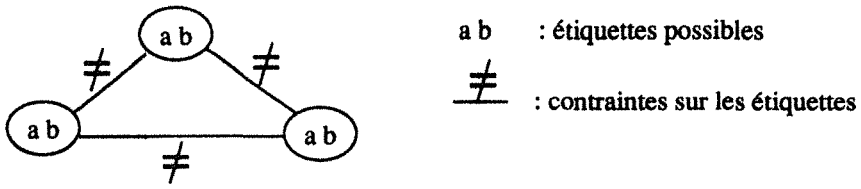


Figure 2.9. *Un réseau de contraintes: les étiquettes doivent toutes être différentes [Mohr 88a].*

Notons que dans le cas réel, les images sont bruitées et par conséquent, une application telle quelle de cet algorithme peut engendrer des éliminations intempestives. Pour remédier à ce problème, il faut assouplir les contraintes afin de pouvoir prendre en compte les erreurs [Mohr 88a]. Ce relachement des contraintes n'est pas toujours facile à mettre en œuvre.

• **Relaxation continue :** Contrairement à la relaxation discrète, les compatibilités sont des valeurs continues. Les problèmes traités par cette technique se présentent sous la forme suivante: classer un ensemble d'objets $\{A_i, i = 1 \ldots N\}$, sur lequel une relation de voisinage est définie, en un ensemble donné de classes ou d'étiquettes possibles $\{L_j, j = 1 \ldots K\}$. Pour chaque A_i, il lui est associé une estimation $p_i(L_j)$ de la distribution de probabilité que A_i ait l'étiquette L_j :

$$(p_i(L_1), \ldots, p_i(L_j), \ldots, p_i(L_K)) \text{ avec } 0 \leq p_i(L_j) \leq 1 \text{ et } \sum_{j=1}^{K} p_i(L_j) = 1$$

Pour tout couple (A_i, A_j) tel que A_i, A_j vérifie la relation de voisinage et pour chaque couple (L_n, L_k), il existe une mesure de compatibilité $r_{ij}(L_n, L_k)$. Cette mesure indique le degré de compatibilité de l'objet A_i assigné à L_n avec A_j assigné à L_k. Un ajustement des probabilités de chaque objet utilisant l'influence du voisinage

et fondé sur ces mesures de compatibilité est réalisé par un processus itératif de mise à jour. Le problème essentiel est de construire les valeurs continues $p_i(L)$ afin de garantir la *convergence* du processus de relaxation qui est difficile à contrôler vers une solution locale satisfaisante. Plusieurs façons permettant de calculer ces probabilités existent déjà [Fekete 81]. A titre d'exemple, nous donnons la règle suivante de mise à jour de ces probabilités de l'itération $(m+1)$ en fonction de celles de l'itération (m) :

$$p_i^{(m+1)}(L) = \frac{p_i^{(m)}(L)(1+q_i^{(m)}(L))}{\sum_{l=1}^{K} p_i^{(m)}(L_l)(1+q_i^{(m)}(L_l))} \quad où$$

$$q_i^{(m)}(L) = \frac{1}{J} \sum_{j=1}^{J} \sum_{l=1}^{K} r_{ij}(L, L_l) p_j^{(m)}(L_l)$$

Notons que J désigne le nombre de voisins de l'objet A_i et que les valeurs r_{ij} peuvent être choisies de façon empirique ou encore respectant le théorème de Bayes dans certains travaux.

Cette méthode a été utilisée par [Faugeras 81, Bhanu 84] pour la mise en correspondance d'objets 2D avec et sans occultation, et par [Ziou 91] pour l'appariement de segments de deux images. Nous pouvons encore citer les travaux de Li *et al.* [Li 92] pour la reconnaissance des images aériennes de réseaux routiers.

2.2.5 En résumé

Nous venons de présenter différentes approches de reconnaissance des formes. Chacune de ces approches possède ces propres points forts et ces propres limitations. Pour profiter des avantages des différentes méthodes, de nombreux travaux sur la possibilité de combiner deux ou plusieurs approches entre elles sont effectués. Ces travaux ont donné naissance aux approches *hybrides* [Bunke 90a, Tsai 90].

Notons que l'utilisation des approches hybrides en reconnaissance de formes n'est pas récente. Par exemple dans de nombreux travaux, les approches statistiques sont combinées avec les approches syntaxiques [Tsai 80, Duerr 79]. Les travaux de Groen *et al.* [Groen 85] sur la reconnaissance de symboles dans un circuit électrique est une application d'une approche hybride combinant les méthodes statistiques avec celles de mise en correspondance structurelle.

Le problème du choix de la méthode à appliquer est un point fondamental pour la suite du travail et dépend du type du problème à résoudre et du domaine d'application. Pour ces raisons, à ce stade de l'étude, nous ne pouvons pas choisir la méthode de reconnaissance sans définir notre problème et nos objectifs. C'est pourquoi, il est nécessaire d'examiner les travaux réalisés en analyse de schémas.

2.3 A propos de l'analyse des schémas

Dans cette partie du chapitre, nous nous focalisons sur l'étude des travaux existants traitant ce problème. Nous exposons de façon détaillée les méthodes d'extraction des composantes constituant le schéma et la reconnaissance des lignes de connexion et des symboles. Notons que nous ne nous intéressons pas à la reconnaissance du texte. Cette présentation est suivie d'une synthèse qui présente les différents points faibles de ces travaux que nous avons tentés de résoudre lors du développement de notre approche.

2.3.1 Présentation de différents travaux

Plusieurs chercheurs [Groen 84, Lin 85, Okazaki 88, Fahn 88, Futatsumata 90, Lee 90] se sont intéressés à l'analyse de schémas. Après une étude bibliographique de ce domaine, nous dégageons trois parties principales constituant un système d'interprétation automatique de schémas : *l'extraction du texte*, *la détection des lignes de connexion* et *la reconnaissance des symboles*. Chaque système possède sa propre philosophie pour réaliser ces objectifs. A présent, un exposé des approches développées auparavant s'impose pour pouvoir se situer par rapport à elles et montrer notre apport dans ce travail.

A- Extraction du texte

Bley [Bley 84] propose une méthode de séparation du texte et du graphique par segmentation de l'image en petites composantes rectangulaires appelées "*composantes primaires*". Le texte est constitué par celles qui forment des composantes connexes de petite taille et le graphique est composé de toutes les autres composantes (Figure 2.10).

Cette idée d'utilisation de la taille des composantes connexes est employée dans la majorité des travaux [Groen 84, Okazaki 88, Futatsumata 90] en fixant un seuil à la taille des composantes de type texte. Par exemple dans [Okazaki 88], la taille maximale d'un caractère est fixé à $4mm \times 4mm$.

Contrairement à ces travaux, Lin *et al.* [Lin 85] extraient le texte à la dernière étape du système. En effet, une vectorisation de toute l'image est réalisée en première étape, puis l'ensemble des segments, non identifés comme des symboles ou des lignes de connexion, sont considérés comme du texte.

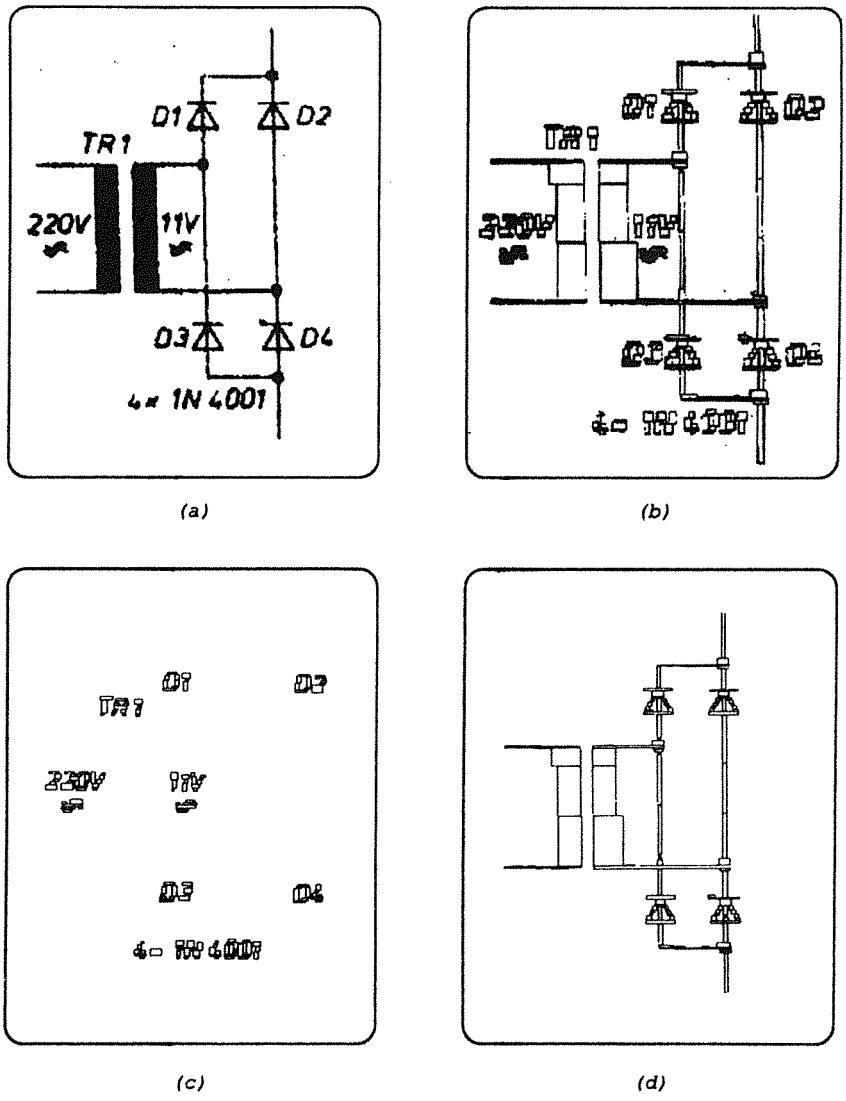


Figure 2.10. *(a) image originale, (b) segmentation en composantes primaires, (c) les composantes connexes formant le texte, (d) schéma après séparation du texte [Bley 84].*

B- Détection des lignes de connexion

Groen *et al.* [Groen 84] proposent de reconnaître en dernier les lignes de connexion. Ils fondent leur raisonnement sur le fait que les symboles et le texte, extraits auparavant, forment le contexte à l'extraction des lignes de connexion.

En revanche, d'autres chercheurs [Lin 85, Okazaki 88, Futatsumata 90] effectuent la reconnaissance des lignes de connexion avant celle des symboles. Pour cela ils utilisent deux hypothèses : les lignes sont supposées horizontales ou verticales, de

plus, elles sont d'une longueur minimale supérieure à un seuil fixé.

Les travaux de Lin *et al.* [Lin 85], utilisent en plus des deux hypothèses citées précédemment, du lien contextuel existant entre les symboles et les lignes de connexion. En effet, toute ligne horizontale ou verticale connectée à une boucle déjà étiquetée comme symbole, mais non encore reconnue, est considérée comme une ligne de connexion.

C- Reconnaissance des symboles

Dans ce paragraphe, nous étudions le problème de localisation et d'identification de symboles. Dans la majorité des travaux, la localisation des symboles est fondée sur l'utilisation des boucles [Groen 84, Lin 85, Okazaki 88]. Toute boucle détectée indique l'existence d'un symbole. Une étape de reconnaissance de ces boucles plus affinée est réalisée dans un deuxième temps. Mais certains symboles, comme la résistance, ne forment pas de boucles et il est donc indispensable de se fonder sur une autre approche pour les localiser. L'hypothèse la plus couramment faite est la suivante : tout ensemble de petits segments connectés entre eux en formant un groupe lui-même connecté à des lignes horizontales ou verticales déjà étiquetées comme lignes de connexion, est un *symbole.*

Lin *et al.* [Lin 85] ne s'intéressent qu'aux symboles contenant des boucles. Toute boucle de longueur supérieure à un seuil est étiquetée comme *symbole.* Cette contrainte de longueur est utilisée pour éviter l'étiquetage d'une boucle du texte comme un symbole puisque Lin *et al.* [Lin 85] séparent le texte à une dernière étape du système. Un repère est défini de façon à ce que son origine soit au centre du rectangle englobant la boucle. Ensuite chaque moitié de boucle est subdivisée en m bandes dans les quatre directions données par la Figure 2.11 ($m = 4$ dans ce cas de figure). Pour chaque bande, une valeur caractéristique est calculée en fonction de la surface de la bande et de la surface de la boucle. Enfin la boucle est définie par le vecteur formé des $4m$ valeurs caractéristiques déjà calculées sur la boucle. Etant donnée une base de boucles modèles définies par des vecteurs caractéristiques calculés de la même manière, la boucle extraite est reconnue comme étant le modèle minimisant la distance euclidienne entre le vecteur caractéristique de la boucle et ceux des modèles des boucles.

Nous trouvons le même principe d'utilisation des boucles pour la prédiction des symboles dans les travaux de Okazaki *et al.* [Okazaki 88]. Les modèles supposés être de même taille, sont classés suivant la forme de leurs boucles et regroupés en *catégories intermédiaires.* La Figure 2.12 montre un exemple de catégorie intermédiaire. Pour chacune de ces classes, ils définissent sa *région minimale d'analyse* qui est la plus petite région pouvant contenir tous les modèles de la catégorie (Figure 2.12).

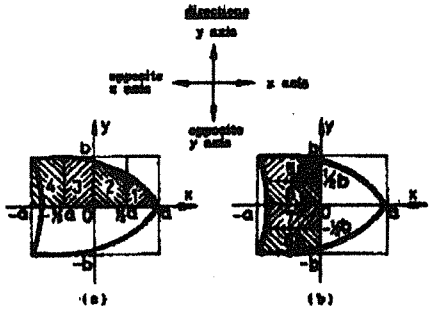


Figure 2.11. *Subdivision des moitiés des boucles en $m = 4$ bandes : (a) dans le sens positif de l'axe des y, (b) dans le sens négatif de l'axe des x [Lin 85].*

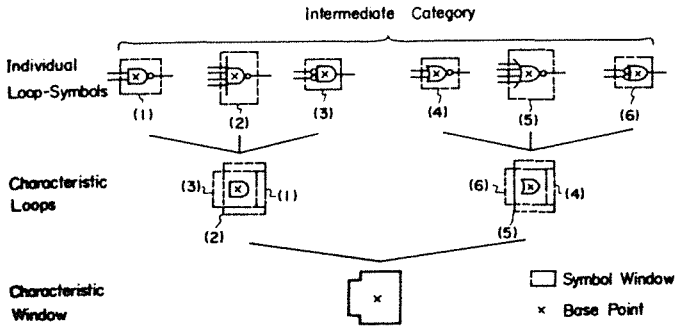


Figure 2.12. *Détermination de la région minimale d'analyse [Okazaki 88].*

Après la séparation du texte, pour chaque boucle détectée dans le schéma, ils déterminent sa catégorie intermédiaire et par conséquent, sa *région minimale d'analyse*. La détermination de la catégorie intermédiaire se fonde sur l'utilisation des caractéristiques intrinsèques de la forme et du rectangle englobant de chacune des boucles. Une analyse plus poussée de la région minimale calculée permet l'identification du symbole au sein de la catégorie intermédiaire trouvée auparavant. Cette analyse est réalisée par une approche hybride. En effet, ils traitent la région deux fois : une première fois pour effectuer un appariement de la boucle contenue dans la zone avec les modèles de boucles de cette catégorie et une deuxième fois par une méthode statistique en utilisant les caractéristiques intrinsèques des modèles. La combinaison des deux résultats se fonde sur un système de règles de production. Les symboles ne contenant pas de boucles sont détectés par le suivi des segments à partir de points caractéristiques (coins ou extrémités) en appariant ces différents points avec le dictionnaire de ces symboles.

Kuner [Kuner 85] fonde son raisonnement sur la recherche d'isomorphismes de

sous-graphe en utilisant des *informations topologiques* des nœuds, c'est-à-dire, le degré des nœuds. Deux nœuds sont mis en correspondance s'ils ont le même degré ; cette égalité permet de résoudre le problème schématisé dans la Figure 2.13. Le graphe représentant tout le schéma est décomposé en sous-graphes ayant la même taille que le modèle courant afin de sélectionner les sous-graphes à apparier. Cette méthode est complétée par l'application d'un ensemble de règles de production [Kuner 86] pour résoudre le problème des images bruitées et les cas ambigus.

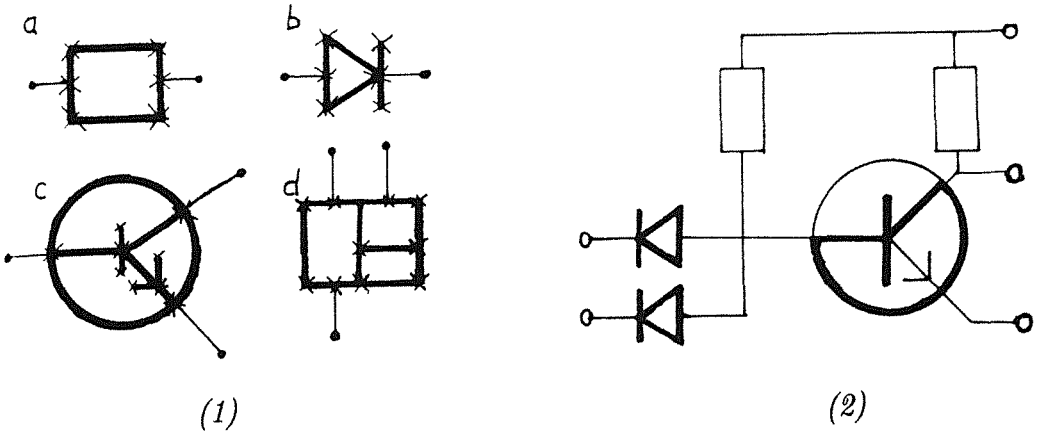


Figure 2.13. *(1) quatre symboles modèles, (2) trois sous graphes appariés avec le modèle b [Kuner 85].*

Groen *et al.* [Groen 85, Lee 90] ont développé une approche hybride combinant une technique statistique et une technique de mise en correspondance. La différence avec les travaux précédents est que les graphes sont attribués et contiennent des informations structurelles et géométriques. Chaque symbole extrait du schéma est apparié avec une base de modèles connus a priori. Une première phase d'apprentissage permet la construction des modèles définis par des informations statistiques. Une estimation de la transformation existante entre le modèle et le symbole est faite en appariant quatre points de confiance du modèle (sélectionnés à la main dans la phase d'apprentissage) avec les quatre points les plus compatibles dans le symbole image. L'image du symbole par la transformation obtenue est ensuite calculée et permet ainsi d'obtenir un symbole normalisé. Puis un appariement entre le modèle et ce symbole normalisé est effectué. Les étapes d'estimation de la transformation et de mise en correspondance sont réalisées avec tous les modèles de la base. Le modèle retenu est celui qui possède le maximum de ressemblance avec le symbole à reconnaître. Cette ressemblance est calculée en employant les informations statistiques propres au modèle et calculées pendant la phase d'apprentissage.

La mise en correspondance est également utilisée pour la reconnaissance des symboles par [Futatsumata 90]. L'analyse des symboles du schéma est réalisée en deux étapes : dans un premier temps une phase de détection des figures caractéris-

tiques (boucles et zones noires) est effectuée ; ensuite l'appariement de ces figures avec une base des figures caractéristiques des modèles est réalisé en considérant que les symboles dans le schéma possèdent la même taille que les modèles. Puis, un appariement des figures auxiliaires (tiges ou petits rectangles collés aux symboles ou appartenant au voisinage de la figure caractéristique) est effectuée. Enfin une phase de vérification des résultats de reconnaissance est effectuée de façon interactive.

2.3.2 Synthèse

Il convient de faire le bilan de cette étude bibliographique afin de dégager les différents points faibles des travaux existants pour pouvoir y remédier. Nous classons ces points en trois catégories ceux qui concernent l'indépendance des processus du système d'interprétation, ceux qui concernent les restrictions faites sur la détection des lignes de connexion et enfin ceux qui concernent la localisation et l'identification des symboles.

A- Indépendance des processus des systèmes d'interprétation de schémas

Le point commun à tous ces travaux, sauf ceux de Lin *et al.*[Lin 85], est que tous ces systèmes possèdent trois processus complètement *indépendants.* Ceci signifie qu'il n'existe aucune *coopération* entre l'extraction du texte, la détection des lignes de connexion et la reconnaissance des symboles bien que ces trois types d'informations soient fortement liés contextuellement. En effet, la présence de texte implique le plus souvent celle d'un symbole au voisinage. Or dans tous les travaux, l'extraction de cette information est faite sans l'utiliser comme guide pour la localisation des symboles. Il est donc évident que le texte n'est pas bien exploité. Cette information peut être particulièrement utilisée pour localiser les symboles ne contenant pas de boucles même si nous ne nous intéressons pas à la reconnaissance du texte. De plus, la reconnaissance des symboles permet d'identifier des lignes comme étant des lignes de connexion et réciproquement les lignes de connexion servent à prédire la présence de symboles à leurs extrémités. Nous constatons que ces liens entre les symboles et les lignes de connexion ne sont pas utilisés dans tous les travaux cités alors que l'on peut supposer que leurs influences sur l'efficacité du système n'est pas négligeable. Remarquons également que Lin *et al.*[Lin 85] ont essayé d'utiliser les symboles pour détecter les lignes de connexion, mais ils n'ont pas bien exploité ce lien, et en aucun cas ils ont pris en compte le lien inverse. En effet, ils ont considéré que les lignes horizontales ou verticales intersectant une boucle sont des lignes de connexion ce qui n'est pas toujours vrai, or après l'étape de reconnaissance nous pouvons détecter de façon exacte les lignes de connexion reliant ce symbole au schéma.

B- Restrictions propres à la détection des lignes de connexion

En plus de ce problème de non communication entre les différentes étapes du système d'interprétation de schémas, nous tenons à signaler qu'il existe d'autres problèmes à examiner. Le premier concerne la restriction des directions des lignes de connexion aux directions horizontale et verticale. Cette restriction est forte puisqu'elle limite le domaine d'application aux circuits électriques où les lignes de connexion sont effectivement horizontales ou verticales et elle exige que le schéma soit numérisé de façon droite. En effet, les lignes de connexion du schéma représenté par la Figure 2.14 ne peuvent pas être reconnues par un tel système.

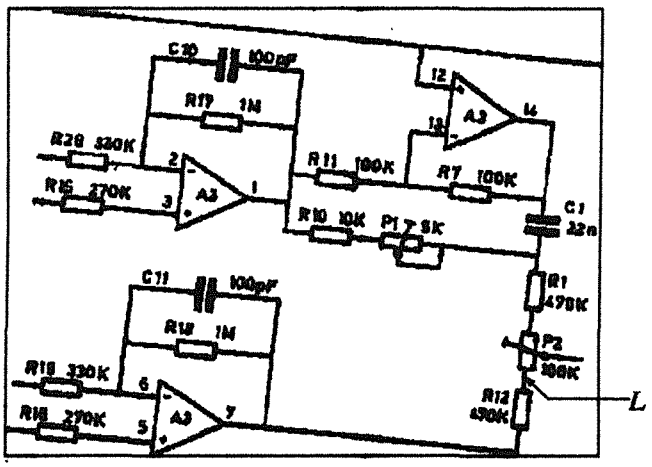


Figure 2.14. *Les lignes de connexion n'appartiennent ni à la direction horizontale ni à la direction verticale.*

Un autre point à signaler concernant la détection des lignes de connexion, est le fait de fixer une valeur minimale à la longueur de ces lignes. Cette autre restriction n'est pas judicieuse puisque cette valeur peut être valable pour un schéma mais pas pour un autre. En plus de ceci et indépendemment de la valeur du seuil, les petites lignes de connexion ne seront jamais détectées, comme par exemple la ligne de connexion L reliant les deux résistances de la Figure 2.14 a une longueur qui est même plus petite que les segments appartenant aux symboles.

Pour toutes ces raisons, la détection des lignes de connexion guidée par le module de reconnaissance de symboles est plus fiable et plus sûre.

C- Problèmes propres à la détection des symboles

En ce qui concerne la localisation des symboles, il est à noter que la majorité des travaux montre un intérêt plus accentué pour la reconnaissance des symboles contenant des boucles, faciles à extraire. Ceci n'est pas suffisant car les symboles qui

ne contiennent pas de boucles (résistance, capacité...) ne peuvent pas être localisés de cette manière. En plus de ceci, notons que l'étiquetage de toutes les boucles comme étant des symboles [Lin 85] engendre beaucoup d'erreurs, puisque l'intersection des lignes de connexion forme plusieurs fausses boucles qui ne correspondent en réalité à aucun symbole (Figure 2.15).

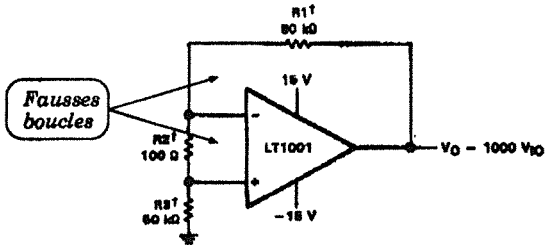


Figure 2.15. *Exemples de fausse boucle.*

D'autres restrictions ont été faites concernant la taille et l'orientation des symboles dans plusieurs travaux. Suivant les approches, certains ne reconnaissent que les symboles dans des directions bien déterminées [Okazaki 88] ou ayant la même taille que le modèle [Kuner 85]. Nous remarquons qu'il est cependant plus intéressant de réaliser la reconnaissance indépendamment de la taille et de l'orientation des symboles comme dans [Lee 90] pour deux raisons :

1. la taille d'un symbole peut changer d'un schéma à un autre et, par conséquent, pour un modèle donné la taille du symbole doit être définie à un facteur d'échelle près,

2. deux occurrences du même symbole peuvent appartenir à un schéma avec deux orientations différentes. Par exemple, le schéma de la Figure 2.16 possède deux occurrences, une horizontale et une verticale, d'une même capacité.

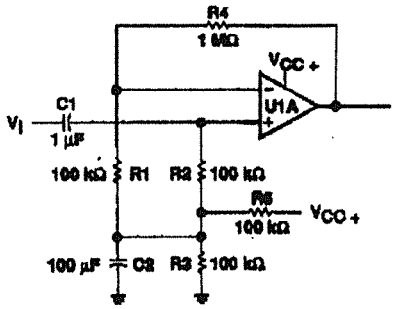


Figure 2.16. *Nécessité de l'invariance de la reconnaissance par rapport la rotation.*

Notons que développer une approche qui estime la valeur de la transformation qui existe entre le symbole et le modèle [Lee 90] n'est pas indispensable puisque cette valeur ne peut pas être utilisée pour normaliser le schéma. En effet, les valeurs de la rotation et de l'homothétie trouvées pour un symbole ne sont pas forcément les mêmes pour un autre symbole.

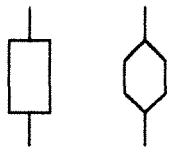


Figure 2.17. *Appariement de graphes fondé sur l'égalité des degrés des nœuds.*

Un autre point concernant la reconnaissance de symboles est que l'information géométrique n'est pas toujours exploitée alors que l'on constate par les travaux de Kuner [Kuner 85] que l'information topologique n'est pas suffisante. En effet, les deux symboles représentés par la Figure 2.17 sont appariés en appliquant l'approche utilisée par Kuner [Kuner 85] bien qu'ils ne représentent pas la même structure géométrique. Nous en déduisons donc que l'utilisation de l'information géométrique est nécessaire au module de reconnaissance.

3

Principes de notre système d'interprétation

Notre objectif principal est d'extraire le texte et de reconnaître toutes les lignes de connexion ainsi que tous les symboles constituant le schéma. Ce chapitre décrit notre système d'analyse de schémas en deux sections. La première section présente la philosophie générale de notre approche (section 3.1), ainsi que les stratégies définies pour chacune des étapes du processus du haut niveau afin de remédier aux différents inconvénients dégagés de l'ensemble des travaux réalisés dans ce domaine. La seconde section expose les processus du bas niveau nous permettant d'extraire les primitives nécessiares à notre travail (section 3.2).

3.1 Philosophie de notre système

Rappelons que les trois composantes, lignes de connexion, symboles et textes, d'un schéma sont *fortement liées* entre elles. L'idée générale de notre système consiste donc à exploiter au mieux ces liens et à utiliser le maximum d'informations disponibles afin de développer un système fiable et efficace qui permet d'éviter les erreurs rencontrées par les systèmes traitant les trois composantes en trois étapes indépendantes (section 2.3).

La Figure 3.1 montre l'architecture générale de notre système. *Le bas niveau* extrait les primitives de bases. *Le haut niveau* interprète le schéma complet et est formé de deux processus :

- *détection des lignes de connexion* est fondée sur l'utilisation du lien contextuel existant entre les informations symboles et lignes de connexion.
- *détection des symboles :* cette détection est fondée sur le principe de *prédiction-vérification des hypothèses* et se décompose en deux modules : la construction des hypothèses de présence de symboles et la reconnaissance de symboles.

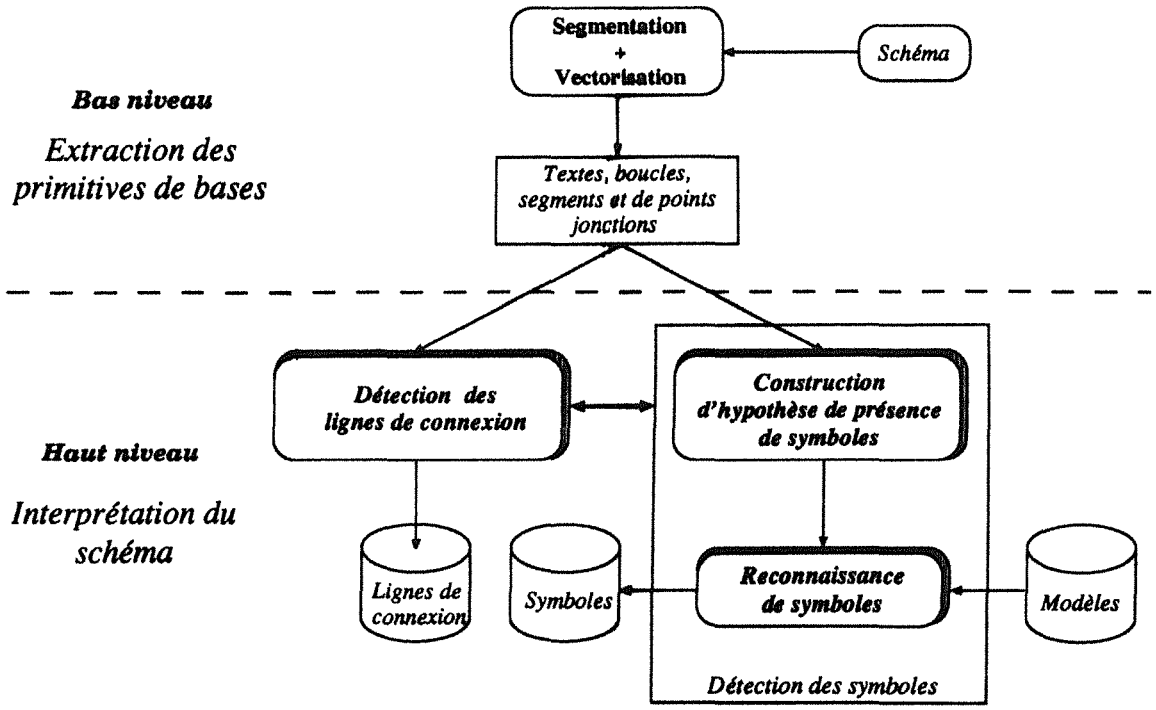


Figure 3.1. *Architecture générale de notre système.*

La construction des hypothèses revient en fait à la sélection des zones susceptibles de contenir les symboles. A la différence avec les différents travaux, nous utilisons en plus des primitives boucles, le texte et les lignes de connexion qui renforcent la probabilité de présence de ces symboles. Nous avons donc choisi d'utiliser toutes ces informations pour sélectionner des zones du schéma susceptibles de contenir des symboles et ainsi constituer les hypothèses de présence de symboles. *La reconnaissance de symboles* vérifie les hypothèses précédentes en identifiant et localisant précisément les symboles contenus dans les zones de présence. Cette étape de vérification des hypothèses permet aussi la détection de nouvelles lignes de connexion à partir des symboles finalement reconnus puisque les symboles sont reliés au schéma par les lignes de connexion.

Les trois modules constituant le haut niveau ont la particularité d'être fortement dépendants l'un de l'autre et ne cessent donc de se communiquer leurs informations : la détection des lignes de connexion est totalement guidée par le résultat de la reconnaissance des symboles dans le cas où l'hypothèse de présence de symbole est confirmée et la reconnaissance des symboles travaille sur les zones sélectionnées par le processus de construction d'hypothèses qui est lui-même fondé sur les résultats de la détection des lignes de connexion. Remarquons qu'à cause de cette dépendance entre la reconnaissance des lignes de connexion et celles des symboles, un traitement

permettant l'activation du système est nécessaire.

Dans la suite de cette section, nous présentons les principes généraux de chacun de ces processus ainsi que les choix que nous avons été amenés à faire.

3.1.1 Détection des lignes de connexion

Le principe de cette étape est fondé sur l'utilisation des liens existants entre les informations symbole et ligne de connexion. En effet, la détection des lignes de connexion est *totalement guidée* par le résultat du module de reconnaissance de symboles. Du moment qu'un symbole est reconnu, les points qui servent à relier le symbole au schéma sont à leur tour reconnu ; nous les appelons *"points de connexion"* (Figure 3.2). A ce moment les lignes du schéma contenant un point reconnu comme point de connexion sont *des lignes de connexion.*

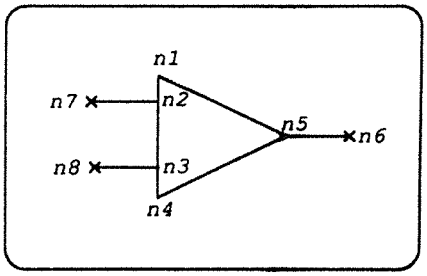


Figure 3.2. n_5, n_6 *et* n_7 *sont les deux points de connexion de l'amplificateur.*

Un des avantages de cette utilisation des liens contextuels est le fait que la détection des lignes de connexion est plus efficace. Notons qu'aucune restriction n'est faite ni sur les directions des lignes de connexion, ni sur leurs longueurs par un seuil donné a priori.

Une étude plus approfondie de ce processus sera faite à la section 5.4 du chapitre 5.

3.1.2 Construction des hypothèses de présence de symbole

Le module de construction d'hypothèses de présence de symbole est lui aussi totalement guidé par les résultats de la détection des lignes de connexion. En effet, il existe une forte probabilité de présence de symboles aux extrémités des lignes de connexion. Cette connaissance est renforcée par les *boucles* ou par la présence du *texte* qui aide à localiser les symboles ne contenant pas de boucles. Nous désignerons, par la suite, *symbole candidat* toute zone susceptible de contenir un symbole sélectionnée par le module de construction des hypothèses de présence de symboles.

Cette utilisation de tous les liens existants entre les trois composantes du schéma possède plusieurs avantages qui s'ajoutent à celui cité dans le paragraphe précédent :

- le fait que la sélection du symbole candidat repose sur l'utilisation des points extrémités des lignes de connexion déjà détectées, permet de minimiser le nombre de fausses boucles à traiter. Contrairement aux travaux précédents, l'utilisation des boucles n'est pas faite à l'aveugle.
- l'utilisation de la présence du texte permet de renforcer la construction des hypothèses de présence pour les symboles ne contenant pas de boucles. Cette notion n'a jamais été utilisée auparavant.

Une étude plus approfondie de ce processus sera faite à la section 5.3 du chapitre 5.

3.1.3 Reconnaissance d'un symbole candidat

Le processus de reconnaissance des symboles candidats cherche à localiser et à identifier le symbole qui peut exister dans une zone délimitée auparavant par le processus de construction des hypothèses. Bien entendu, la zone sélectionnée peut ne pas contenir de symboles et par conséquent la reconnaissance échoue, ce qui entraine une remise en cause de l'hypothèse de présence de symbole dans cette zone.

Pour reconnaître un symbole candidat, nous utilisons une base de modèles connus a priori. La Figure 3.3 montre un échantillon de modèles que nous manipulons. Les points de connexion de chacun des modèles sont connus et marqués par une croix.

Notre but est de trouver le modèle contenu dans la zone de présence. En d'autres termes c'est trouver le modèle ayant la même structure topologique et géométrique que celle d'une sous partie de la zone candidate, indépendamment de la taille et de l'orientation du modèle par rapport à celle du symbole candidat. Rappelons que la reconnaissance des points de connexion est un point important dans notre travail, par conséquent nous souhaitons avoir comme résultat du module de reconnaissance de symboles une *description exacte* de la partie qui sera reconnue, ce qui veut dire une correspondance point à point entre le modèle et cette partie afin d'identifier les points de connexion et par conséquent, les lignes de connexion reliant ce symbole au schéma.

Notons que la taille de la base des modèles peut être importante ; de ce fait il faut avoir une approche qui tienne compte de cette information et qui minimise la complexité de l'algorithme. Nous devons aussi tenir compte du bruit introduit par la numérisation du schéma et le traitement du bas niveau (segmentation, approximation polygonale et vectorisation). Par exemple à la Figure 3.4, il existe des segments manquants alors que d'autres sont subdivisés en petits segments et de

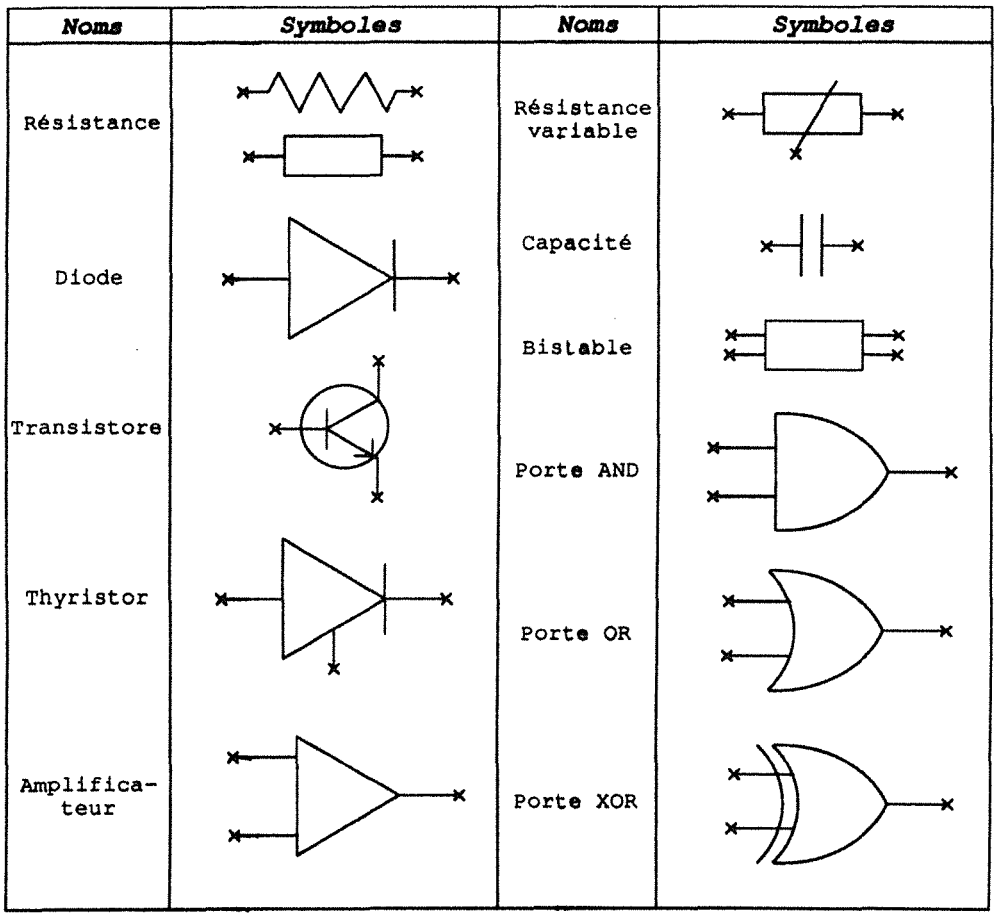


Figure 3.3. *Exemples de symboles modèles.*

plus des points de jonctions qui sont non détectées. Il ne faut pas oublier également les erreurs que l'étape de construction des hypothèses de présence des symboles ne donne pas un symbole parfait mais elle fournit une zone susceptible de contenir le symbole à reconnaître. En effet, le symbole candidat de la Figure 3.4 contient de nombreux segments n'appartenant pas à la résistance

A ce stade de l'étude, la première question que nous nous posons est la suivante: *Quelle méthode de reconnaissance, parmi celles présentées dans le chapitre précédent, répond au mieux à nos objectifs ?* Il est évident qu'il n'existe pas de critères permettant d'établir le choix idéal de telle ou telle méthode suivant le type de problème posé. Pour notre part les approches structurelles (§ 2.2.3 et § 2.2.4) nous paraissent les plus adaptées à notre problème dans le sens qu'elles donnent directement en résultat une description précise de la forme, en l'occurence du symbole et elles manipulent les informations structurelles des formes or les symboles que nous traitons sont des formes où l'information structurelle est *dominante*. Les

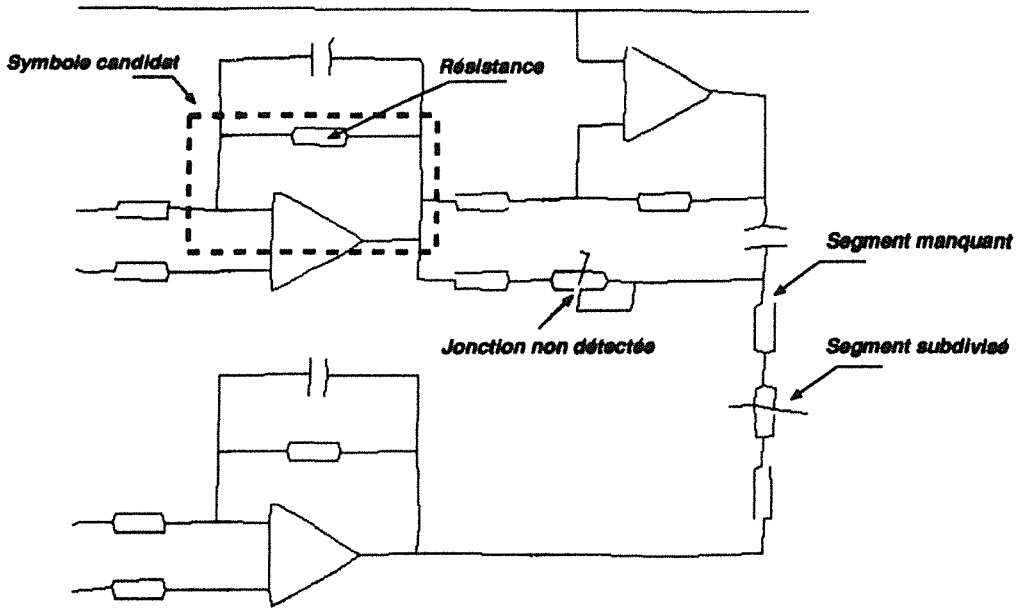


Figure 3.4. *Exemples d'erreurs affectant le schéma.*

approches structurelles se partagent en deux groupes, d'une part les techniques de mise en correspondance et d'atres part les techniques syntaxiques. Parmi ces deux types d'approches, *la mise en correspondance* de graphes nous parait plus appropriée pour notre problème que l'approche syntaxique puisque les approches syntaxiques manipulent très bien les ensembles *infinis de modèles* et sont préférées dans le cas où les modèles peuvent être hiérarchiquement classés [Bunke 90a]. Mais ce n'est pas notre cas et de plus, l'approche syntaxique est mal adaptée aux images bruitées. Par contre la mise en correspondance est plus adaptée dans le cas où l'ensemble des modèles est *fini*.

En résumé, nous avons développé une approche fondée sur *la mise en correspondance structurelle*. Une fois l'appariement réalisé avec tous les modèles, une étape de choix du meilleur modèle est alors effectuée. Cette mise en correspondance est effectuée en absence de toute information concernant la transformation (rotation ou homothétie) qui existe entre le symbole candidat et le modèle. L'originalité de cette approche réside dans l'utilisation d'un maximum d'informations *structurelles*. Nous tenons à signaler que nous procédons à une vérification du résultat de l'appariement en nous reposant sur des informations contextuelles (chapitre 4). Ce processus de reconnaissance de symboles candidats fait l'objet du chapitre suivant.

3.2 Extraction des primitives de bases

Pour pouvoir interpréter un document[1], il faut construire des primitives plus évoluées que le pixel, c'est à dire renfermant une information plus pertinente. Les processus de construction de ces entités (segments, points de jonctions, composantes connexes...) constituent *le bas niveau.*

Cette section introduit les différentes étapes d'un processus de bas niveau qui nous permet d'extraire les primitives de bases dont nous avons besoin ; ceci consiste à localiser le *texte*, les *boucles* et à décrire l'image en termes de *segments* et *points.* Nous commençons par une présentation des principales techniques de vectorisation (§ 3.2.1). Ensuite, la structure générale de notre processus de bas niveau est exposée en 3.2.2. L'extraction du texte d'un document composite fait l'objet du paragraphe 3.2.3. Enfin nous étudions la détection des boucles caractéristiques (§ 3.2.4).

3.2.1 Principales techniques de vectorisation

Le principal processus de bas niveau convertit l'image en un ensemble de vecteurs. Les deux techniques les plus répandues sont *la squelettisation* d'une part et *l'appariement de contours* d'autre part. La présentation de ces deux techniques est suivie d'une brève présentation d'autres techniques originales utilisées pour vectoriser les circuits électriques.

A- Technique de squelettisation

La *squelettisation* est une méthode souvent utilisée. Les algorithmes de squelettisation habituellement employés procèdent par amincissement successifs sans affecter la connexité du graphique. Les propriétés fondamentales de ces techniques sont, d'une part, la conservation de la connexité de l'image et d'autre part, la détermination des axes médians des composantes connexes. Néanmoins ce type d'approches pose quelques problèmes comme l'apparition des barbules ou l'instabilité du traitement des points de jonction. Le squelette étant formé d'un ensemble de pixels, l'étape suivante effectue une approximation polygonale de ce squelette pour fournir l'ensemble des segments se joignant en des points de jonction. Parmi les algorithmes connus, nous pouvons citer celui de Harris [Harris 82] qui présente un algorithme de squelettisation suivie d'une approximation polygonale avec une application de cet algorithme pour la reconnaissance de caractères.

L'approche par squelette a aussi été utilisée par Groen *et al.* [Groen 84] pour l'analyse des circuits électriques, mais également, par Landy [Landy 85] et Bap-

1. Les documents traités sont saisis par un scanner.

tista [Baptista 88] pour la reconnaissance de l'écriture manuscrite des nombres.

La squelettisation permet de connaître uniquement la structure des formes ne contenant pas de zones noires et par conséquent ce type d'approche n'est pas bien adapté aux schémas que nous traitons (Figure 3.5). Aussi d'autres techniques de vectorisation telle que l'appariement de contours [Cugini 84] se proposent-elles d'éliminer cet inconvénient.

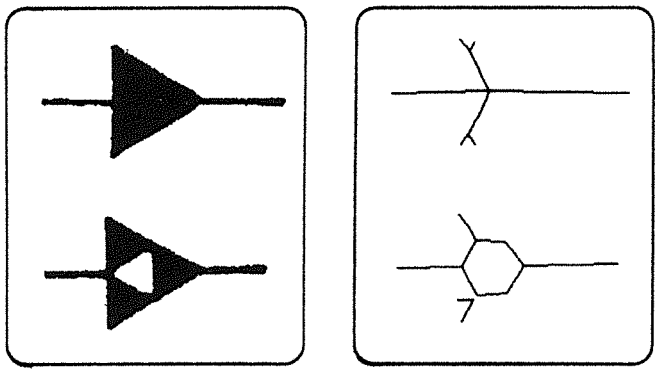


Figure 3.5. *Inadaptation de la squelettisation :(a) Images originales, (b) les deux symboles sont complètement déformés après l'application d'un algorithme de squelettisation.*

B- Technique d'appariement de contours

Une autre technique de vectorisation apparie les contours opposées des composantes connexes, ce qui nécessite une segmentation en composantes connexes, réalisée au préalable. Les travaux de Cugini [Cugini 84] et ceux de Tombre [Tombre 87] se fondent sur ce principe. Ces derniers ont conduit à la naissance du système REDRAW[2] développé dans notre équipe par Karl Tombre [Tombre 87, Antoine 90]. Trois étapes constituent ce système :

- *La segmentation :* Dans un premier temps l'image est segmentée en composantes connexes constituant une structure arborescente d'inclusion des composantes connexes blanches et noires.

- *L'approximation polygonale :* La phase de segmentation est suivie par une approximation polygonale appliquée sur les chaînes de contours (contour interne et externe) des composantes connexes : l'algorithme utilisé est celui de Wall et Danielsson [Wall 84] et fournit comme résultat un ensemble de segments.

2. REading DRAWings.

- *La vectorisation :* Elle apparie deux segments (l'un est en face de l'autre) pour fournir un seul segment traduisant les traits dans le document. Cet ensemble de segments est accompagné d'une structure de points de jonction.

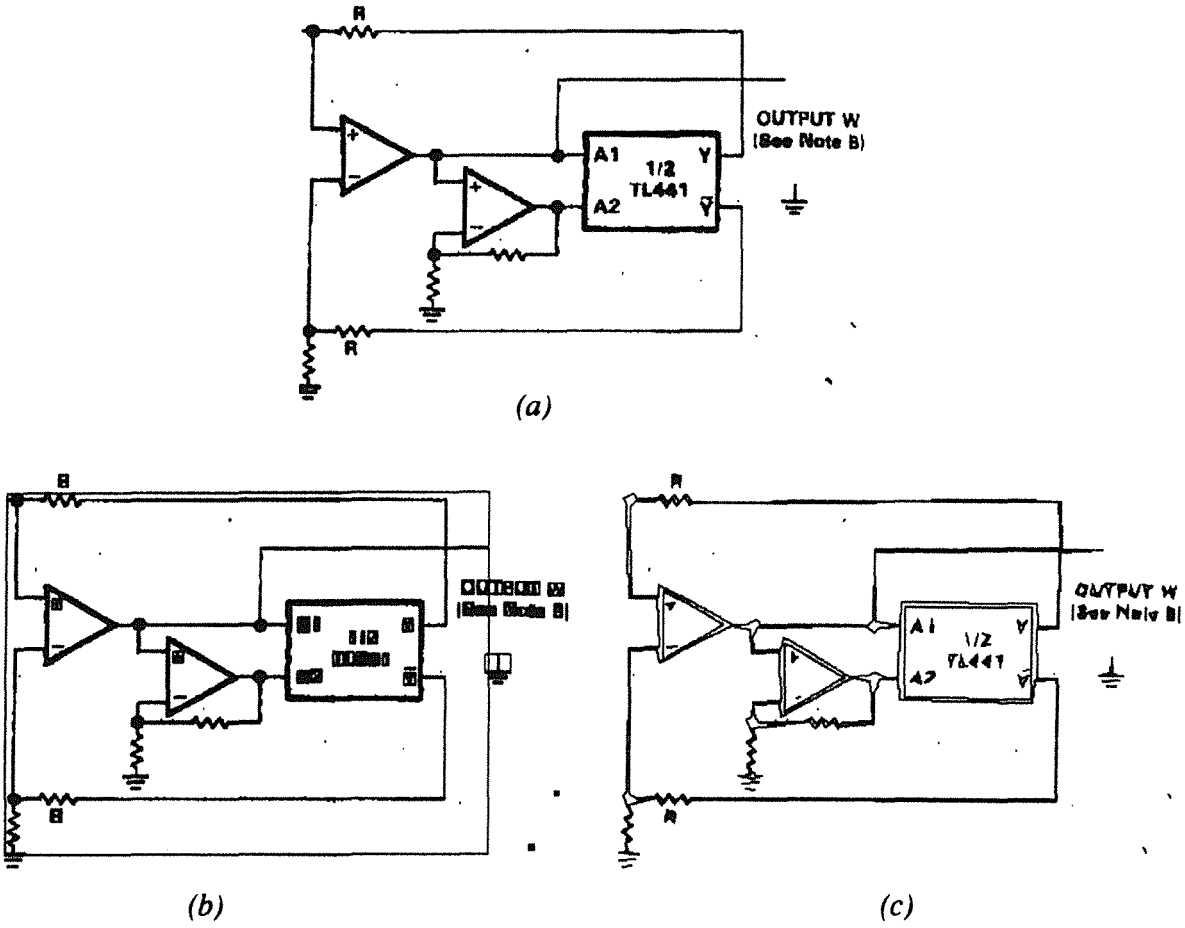


Figure 3.6. *(a) image originale, (b) résultat de la phase de segmentation : chaque composante connexe noire est représentée par son rectangle englobant, (c) résultat de l'approximation polygonale des contours internes et externes.*

La Figure 3.6 montre un exemple d'application des deux premières étapes du système REDRAW sur un circuit électrique. Ce système comporte une méthode bien adaptée aux traitements des documents techniques et plus particulièrement les composantes graphiques puisqu'il remédie aux problèmes posés par la squelettisation. En effet, l'utilisation de la notion de contour interne et externe permet la vectorisation des symboles contenant des zones noires. La Figure 3.7 montre le résultat de l'étape d'approximation polygonale de REDRAW sur les images de la Figure 3.5. Notons que contrairement aux techniques de squelettisation, dans ce cas la forme globale du symbole n'est pas affectée. Remarquons que les petites taches à

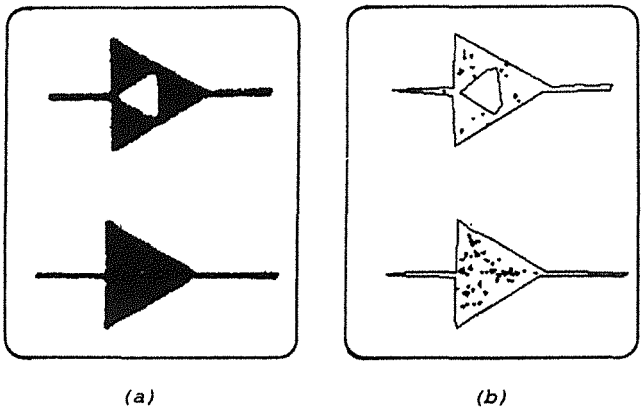


Figure 3.7. *(a) images originales, (b) résultat de l'approximation polygonale des contours internes et externes.*

l'intérieur des symboles peuvent être éliminées par une simple dilatation des images avant d'appliquer REDRAW.

C- Autres techniques

Evidemment les deux techniques que nous venons de présenter ne sont pas les seules à être utilisées pour vectoriser les documents techniques. Par exemple Lin *et al.* [Lin 85] proposent de diviser l'image en mailles de taille appropriée: l'épaisseur maximale d'une ligne et la distance minimale entre deux lignes sont supposées connues. Les formes caractéristiques sont déterminées en étudiant les intersections des lignes de l'image avec les bords de chaque maille (Figure 3.8). Ces formes reflètent les caractéristiques structurelles locales de l'image.

Un étiquetage de l'image conformément aux étiquettes des formes caractéristiques est réalisé en balayant l'image. Une étude de cet étiquetage permet la détermination des différentes lignes constituant le schéma. Par exemple une suite de "I" est transformée en une ligne verticale et une suite de "-" en une ligne horizontale. Une étude plus fine est réalisée pour la détection des points caractéristiques où l'étiquetage par aucune des formes n'est possible.

Bley [Bley 84] propose également de vectoriser les images de circuits électriques en segmentant l'image en composantes primaires (§ 2.3.1), puis en détectant les lignes dominantes (assez longues) dans les directions horizontale, verticale ou diagonale en utilisant la relation d'adjacence des différentes composantes primaires (Figure 3.9).

Les détails qui ne vérifient pas les critères d'adjacence sont traités par un système

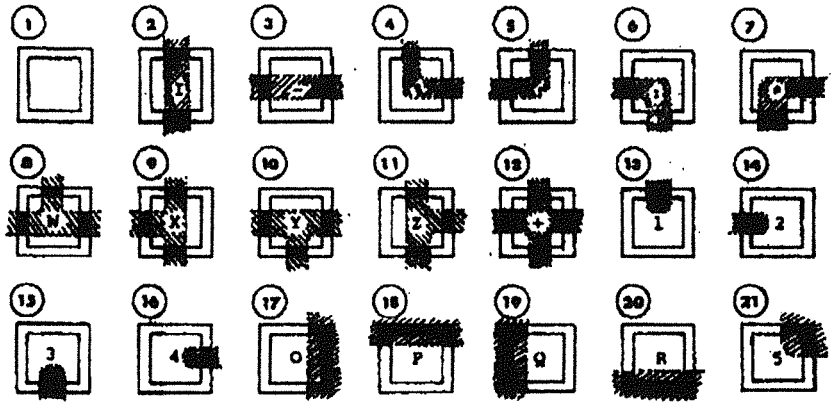


Figure 3.8. *Exemples de formes caractéristiques; il existe 51 formes en total. Chaque forme possède sa propre étiquette (par exemple la forme (2) est étiquetée par le caractère "I") [Lin 85].*

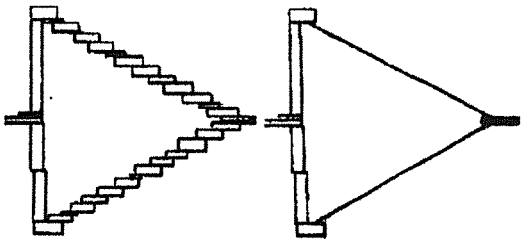


Figure 3.9. *Exemple de détection des lignes diagonales à partir des composantes primaires [Bley 84].*

de règles de productions spécifiques aux circuits électriques. La configuration de la partie gauche de la règle de production donnée par la Figure 3.10 ne peut pas être traitée uniquement par la relation d'adjacence. En effet, l'adjacence des deux composantes primaires n'est pas vérifiée puisque il existe un segment horizontal qui les sépare. Ce cas problématique est traité par cette règle de production (Figure 3.10).

3.2.2 Structure globale du processus du bas niveau

La vectorisation par appariement des contours est celle qui fournit la meilleure qualité de résultat. Nous avons donc utiliser dans un premier temps le système REDRAW. Mais dans l'état actuel, l'étape de vectorisation de REDRAW n'est pas encore opérationnelle. De ce fait, pour pouvoir valider notre système d'analyse de schémas sur des images réelles, nous avons utilisé un algorithme de squelettisation

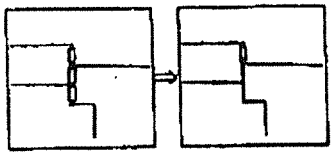


Figure 3.10. *Exemple de règles de production [Bley 84].*

suivie d'une approximation polygonale disponible dans notre laboratoire. Ces deux outils sont intégrés dans le logiciel VISILOG[3]. Un exemple d'application de ces deux outils est donné par la Figure 3.11. Remarquons que la squelettisation ne doit être appliquée que sur la partie graphique du schéma, ce type de vectorisation n'était pas adapté au texte.

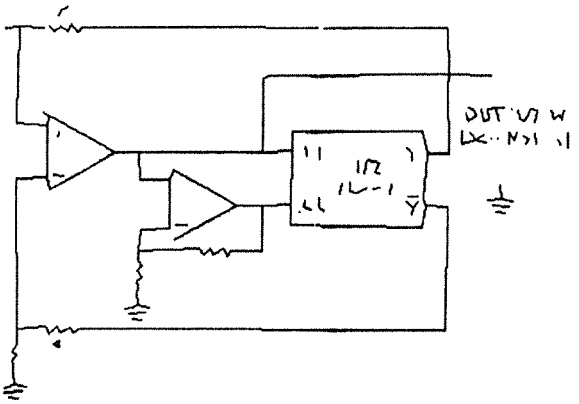


Figure 3.11. *Résulat de l'algorithme de la squelettisation suivi par une approximation polygonale appliqué sur l'image de la Figure 3.6.a.*

Notre objectif actuel consiste non seulement à vectoriser l'image mais aussi à localiser le texte et à détecter les boucles, or il ne suffit pas d'appliquer uniquement un algorithme de squelettisation suivi d'une approximation polygonale sur l'image binaire. La Figure 3.12 montre la structure générale de notre bas niveau permettant l'extraction des primitives de bases : *textes*, *boucles* , *segments* et *points de jonctions.* L'extraction du texte est fondée sur la taille des composantes connexes (§ 2.3.1). Pour cela, une étape de segmentation du schéma en composantes connexes est faite. Ce module est celui du système REDRAW. La squelettisation et l'approximation polygonale sont du logiciel VISILOG. Les modules de séparation du texte et la détection des boucles caractéristiques font l'objet des deux sections suivantes.

3. Boite à outils de traitements d'images réalisée par NOESIS.

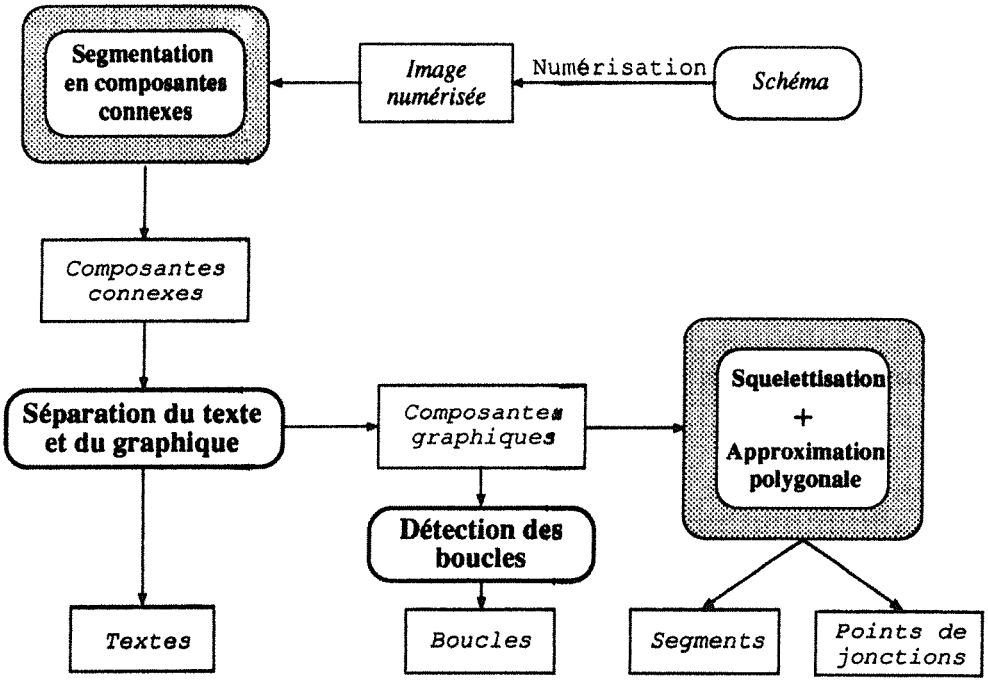


Figure 3.12. *Notre processus de bas niveau.*

3.2.3 Extraction du texte d'un document composite

Nous avons vu dans l'étude des différents travaux faits sur l'interprétation de schémas (Section 2.3.1) que la localisation du texte est fondée sur la taille des composantes connexes [Bley 84, Groen 84, Okazaki 88, Futatsumata 90], en fixant un seuil limite à la taille des caractères (par exemple dans [Okazaki 88] la taille maximale d'un caractère est de $4mm \times 4mm$). Ceci est assez contraignant du fait que la valeur du seuil peut être bonne pour un schéma mais pas pour un autre schéma.

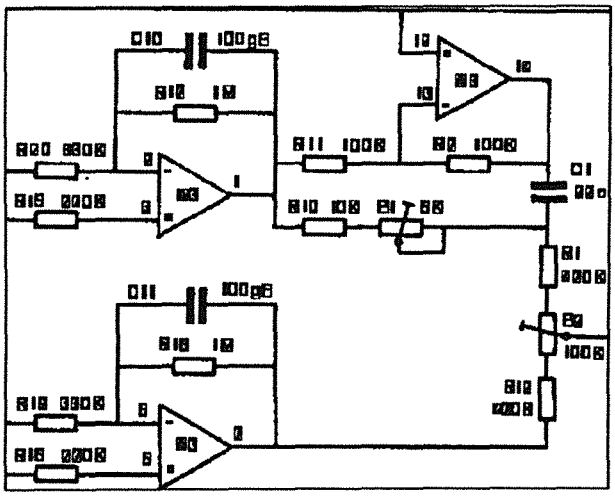
Les travaux de Fletcher et Kasturi [Fletcher 88] remédient à ce problème en proposant un algorithme de séparation du texte et du graphique sans aucune restriction ni sur la taille des caractères, ni sur la police, ni sur l'orientation. La segmentation est réalisée en fonction de la taille *relative* des composantes textes par rapport aux composantes graphiques ; ceci est possible du fait que le texte est de taille très inférieur à celles du graphique. L'idée consiste à calculer un histogramme déterminant la fréquence relative des occurences des composantes connexes noires en fonction de leurs surfaces. Cet histogramme permet de partitionner l'ensemble des composantes connexes noires en deux classes texte et graphique. Le seuil sur la taille des composantes textes est alors déterminé automatiquement à partir de cet histogramme. Nous avons utilisé ce principe pour réaliser la séparation entre le texte et le graphique.

Après numérisation et segmentation, il est évident que deux caractères de même taille peuvent se voir associer, deux composantes connexes de taille légèrement différentes. Cette différence est due au bruit et d'après nos expérimentations, nous estimons la dégradation de l'ordre de 10% près. Dans le calcul du seuil ces deux composantes votent pour une composante de taille égale à la moyenne de leurs tailles, ce qui veut dire qu'un réajustement de l'histogramme est réalisé au fur et à mesure de la mise à jour de l'histogramme pour chaque nouvelle composante connexe. L'algorithme suivant réalise cette mise à jour :

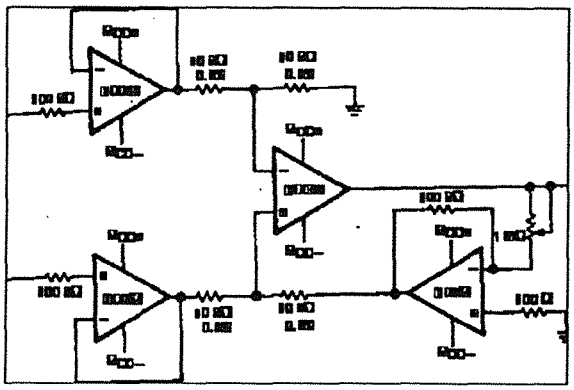
Fonction *Calcul_histogramme(histogramme,comp-connex)*
histogramme : une liste de couple (surface,frequence)
comp-connex : nouvelle composante connexe à intégrer dans l'histogramme
surface ← *surface(comp-connex)*
/* A ce moment la fréquence de cette composante est initialisée à 1 */
frequence ← *1*
reajustement_hist(histogramme,surface,frequence)
Fin Fonction

Fonction *reajustement_hist(histogramme,surface,frequence)*
/* Fonction qui réalise la mise à jour de l'histogramme */
$(hist_1, hist_2)$ ← *meme_surface(histogramme,surface,10%)*
/* *meme_surface* détermine les surfaces ayant la même taille que *surface* à 10% près */
Si $(hist_1 \neq NULL) and (hist_2 \neq NULL)$
Alors /* Surface comprise entre deux valeurs de l'histogramme et elle est la meme que ces deux valeurs à 10% */
$surface \leftarrow (surface + hist_1.surface + hist_2.surface)/3$
$frequence \leftarrow frequence + hist_1.frequence + hist_2.frequence$
$histogramme \leftarrow histogramme - \{hist_1 \cup hist_2\}$
reajustement_hist(histogramme,surface,frequence)
Sinon
Si $(hist_1 \neq NULL)$
Alors/* une seule valeur de l'histogramme est équivalente à surface */
$surface \leftarrow (surface + hist_1.surface)/2$
$frequence \leftarrow frequence + hist_1.frequence$
$histogramme \leftarrow histogramme - hist_1$
reajustement_hist(histogramme,surface,frequence)
Sinon
$histogramme \leftarrow histogramme \cup (surface, frequence)$
Fin Fonction

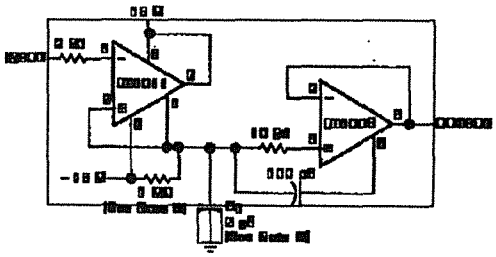
Le pic de l'histogramme ne peut correspondre qu'à des composantes texte du fait que celles-ci sont plus nombreuses que les composantes graphiques[4]. La valeur S_{moy}, correspondant au pic de l'histogramme, estime la taille moyenne des caractères. Mais les caractères pouvant être collés, 3 par 3 au maximum[5], nous définissons *la surface de séparation* S_{sep} comme étant $3 * S_{moy}$. La séparation est donc réalisée de la façon suivante : toute composante de taille inférieure à S_{sep} est *une composante texte.*



(a) Schéma 1



(b) Schéma 2



(c) Schéma 3

Figure 3.13. *Segmentation en composantes connexes.*

La Figure 3.13 montre le résultat de la segmentation en composantes connexes. Nous constatons qu'effectivement la taille d'une composante texte est très petite par rapport à celle d'une composante graphique et que dans tous ces schémas, le nombre de composantes connexes texte est beaucoup plus élevé que celui des composantes graphiques.

Les histogrammes obtenus pour les schémas de la Figure 3.13 sont donnés par le tableau suivant :

4. Dans la majorité des schémas, il existe une seule composante graphique.
5. Valeur chosie pour ce type de documents.

Figure 3.13.a		Figure 3.13.b		Figure 3.13.c	
Surface	*Fréquence*	*Surface*	*Fréquence*	*Surface*	*Fréquence*
12	2	12	1	14	2
18	2	16	6	16	1
34	20	18	1	22	4
42	1	24	2	31	11
47	9	32	2	35	2
56	1	36	19	42	6
70	5	**53***	**28***	49	13
84	5	63	14	**62***	**26***
94*	**48***	72	9	71	15
109	36	81	13	80	4
132	1	100	6	95	2
234686	1	127	3	704	1
		166478	1	56274	1

Les valeurs marquées par * correspondent aux pics des histogrammes de chacune des images. Nous constatons que la séparation est immédiate. Par exemple pour la Figure 3.13.a, nous obtenons $S_{moy} = 94$ et par conséquent, la surface de séparation S_{sep} est égale à 282 et toute composante de taille inférieur à 282 est une composante texte.

La Figure 3.14 montre les composantes graphiques ainsi que les composantes textes des trois schémas de la Figure 3.13 après séparation du texte et du graphique.

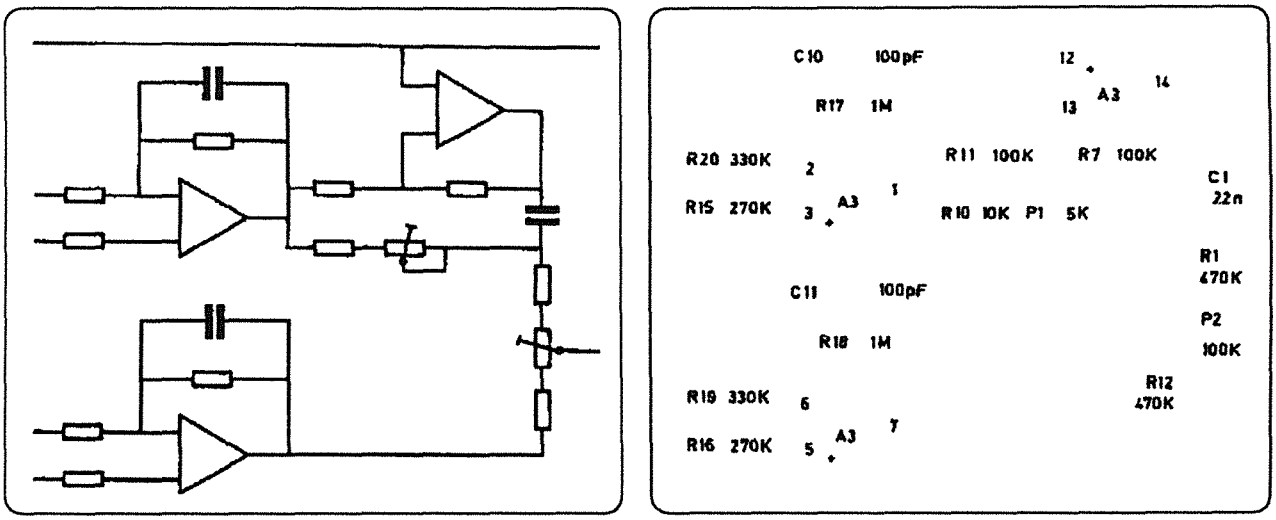
3.2.4 Détection des boucles caractéristiques

Une boucle est définie par un contour fermé. La détection des boucles est fondée sur les *composantes connexes blanches.* Toute composante blanche (à part la composante du fond de l'image) est une boucle. La Figure 3.15 montre trois types de boucles : les boucles *(1)* et *(2)* appartiennent à du graphique, par contre la boucle *(3)* appartient à du texte. Pour cette raison, nous remarquons que toutes les boucles d'un schéma ne sont pas significatives pour guider la construction d'hypothèse de présence de symbole.

Nous définissons alors ce que nous appelons *une boucle caractéristique* comme étant une composante connexe blanche appartenant à du graphique et directement incluse dans la composante blanche contenant tout le schéma[6]. En effet, dans la Figure 3.15, la composante blanche *(1)* est une boucle caractéristique puisqu'elle appartient à du graphique et est directement incluse dans le fond de l'image ; en revanche les composantes *(2)* et *(3)* ne le sont pas puisque la composante *(2)* est incluse dans la composante *(1)* et la composante *(3)* appartient à du texte. Notons que les boucles caractéristiques sont définies par leurs rectangles englobants.

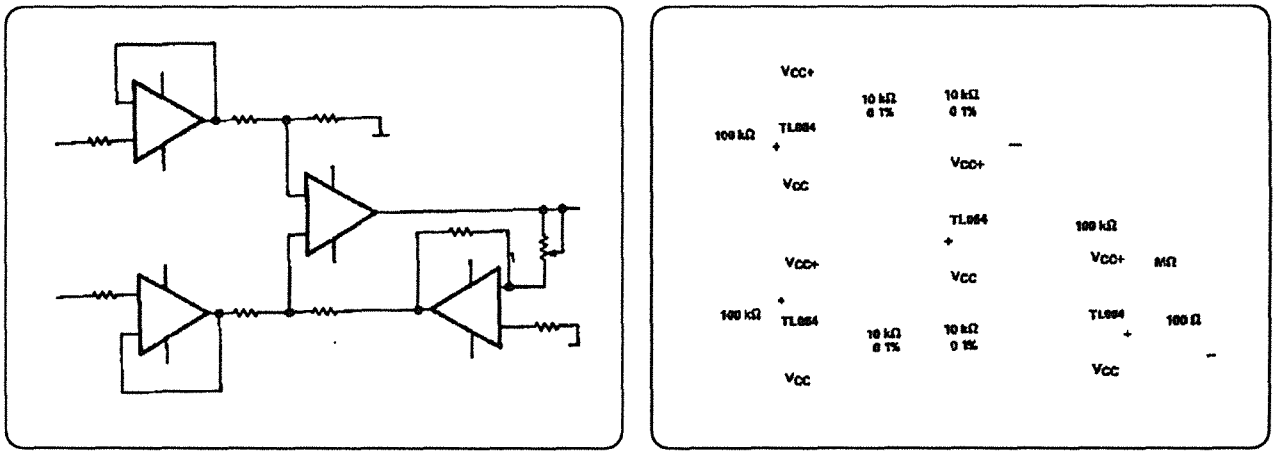
La détection des boucles caractéristiques est alors effectuée une fois la séparation du texte réalisée pour éliminer toutes les boucles appartenant à du texte et elle utilise la structure arborescente d'inclusion des composantes connexes fournie par la phase

6. En général, cette composante constitue le fond de l'image.



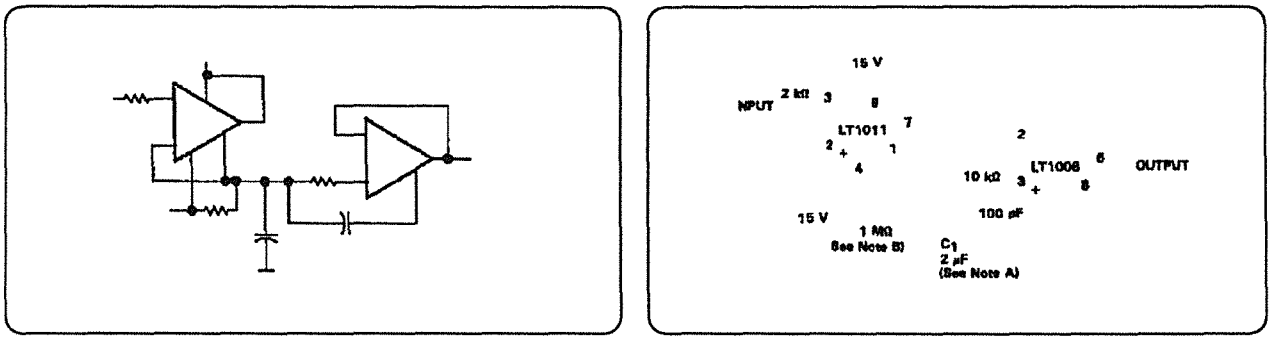
(a) *(b)*

Schéma 1



(a) *(b)*

Schéma 2



(a) *(b)*

Schéma 3

Figure 3.14. *Séparation du texte et du graphique : (a) composantes graphiques des schémas de la Figure 3.13, (b) composantes textes des schémas de la Figure 3.13.*

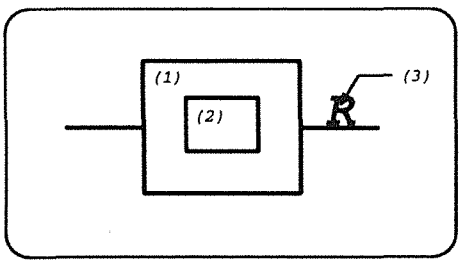


Figure 3.15. *Exemple de boucles.*

de segmentation pour déterminer celles qui sont directement incluses dans la plus grande composante blanche contenant tout le schéma.

La Figure 3.16 montre deux exemples de schémas avec leurs arborescences respectives construites avant que la séparation du texte soit effectuée. Dans la Figure 3.16.a, N_1 et N_2 sont les deux composantes graphiques du schéma. Dans la Figure 3.16.b, $N_2 \ldots N_{p'}$ sont du texte; N_{p+1}, N_{p+2} et N_{p+3} inclus dans B_1 correspondent au 3 caractères 15V et les $N_{p+4}, \ldots, N_{p'}$ correspondent aux caractères LT1001, +, - inclus dans la boucle B_3.

La détection des boucles caractéristiques revient à déterminer les composantes du deuxième niveau de la partie de l'arborescence qui correspond au graphique. En effet, dans la Figure 3.16.a, seule la composante blanche B_1 est une boucle caractéristique. Alors que dans la Figure 3.16.b, il existe trois boucles caractéristiques qui sont B_1, B_2 et B_3.

La Figure 3.17 montre un exemple de résultat de détection des boucles caractéristiques. Par exemple, les boucles A et B sont détectées comme boucles caractéristiques et les rectangles englobants A' et B' correspondent respectivement à A et B. Notons que nous pouvons obtenir des fausses boucles par exemple la boucle B ne correspond à aucun symbole et par conséquent, nous devons en tenir compte par la suite lors de son utilisation.

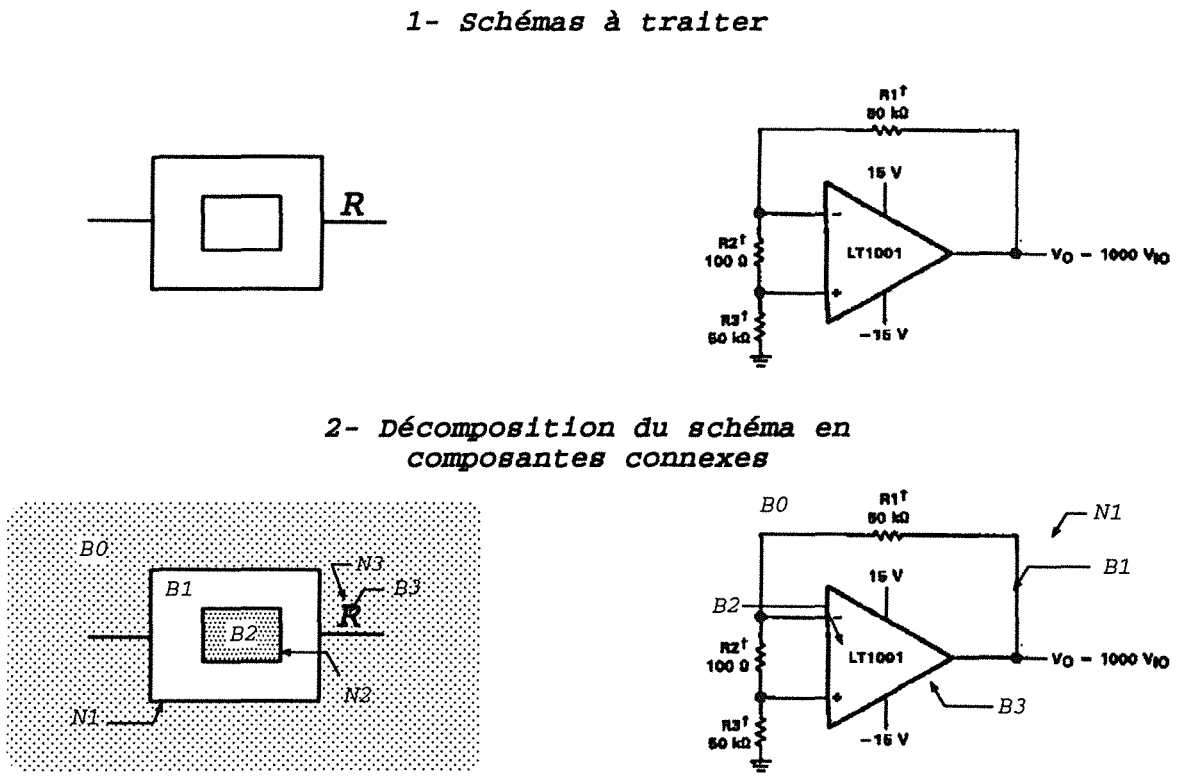


Figure 3.16. *Les B_i représentent les composantes connexes blanches et les N_i représentent celles qui sont noires. B_0 correspond au fond de l'image. (a) B_1 est une boucle caractéristiques, (b) B_1, B_2 et B_3 sont des boucles caractéristiques.*

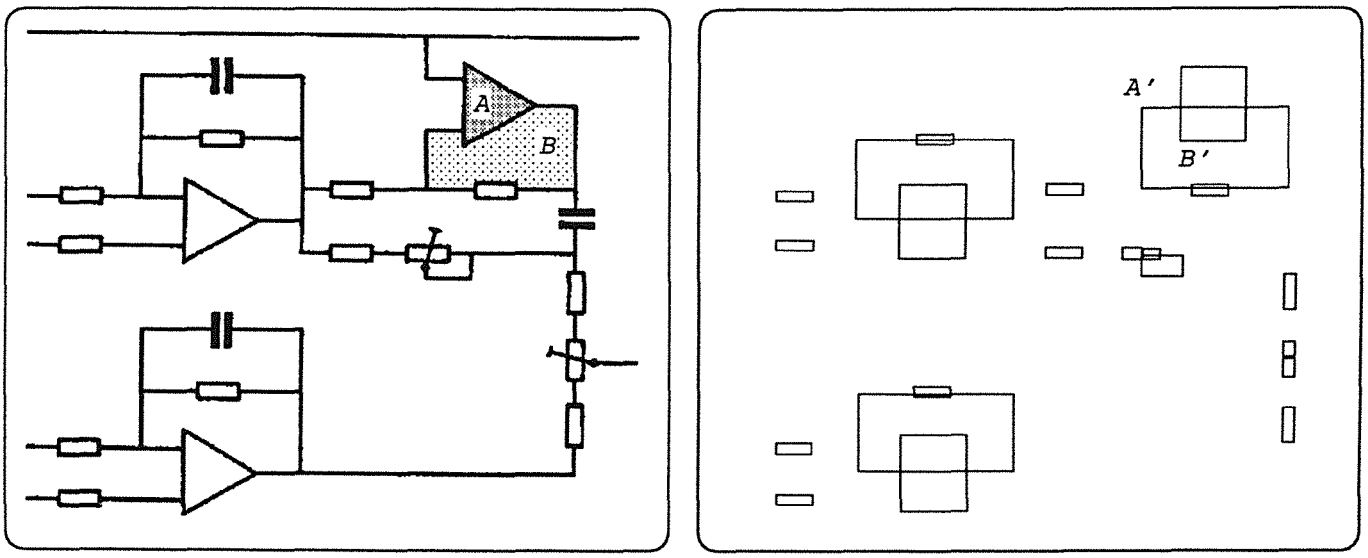


Figure 3.17. *(b) Ensemble des rectangles englobants des boucles caractéristiques du schéma électrique de (a).*

4
Reconnaissance d'un symbole candidat

Nous rappelons que la reconnaissance est fondée sur la mise en correspondance structurelle. Une fois la mise en correspondance du symbole candidat avec tous les modèles de la base effectuée, le choix du meilleur modèle apparié avec le candidat est réalisé. La Figure 4.1 montre l'architecture générale du processus de reconnaissance d'un symbole candidat.

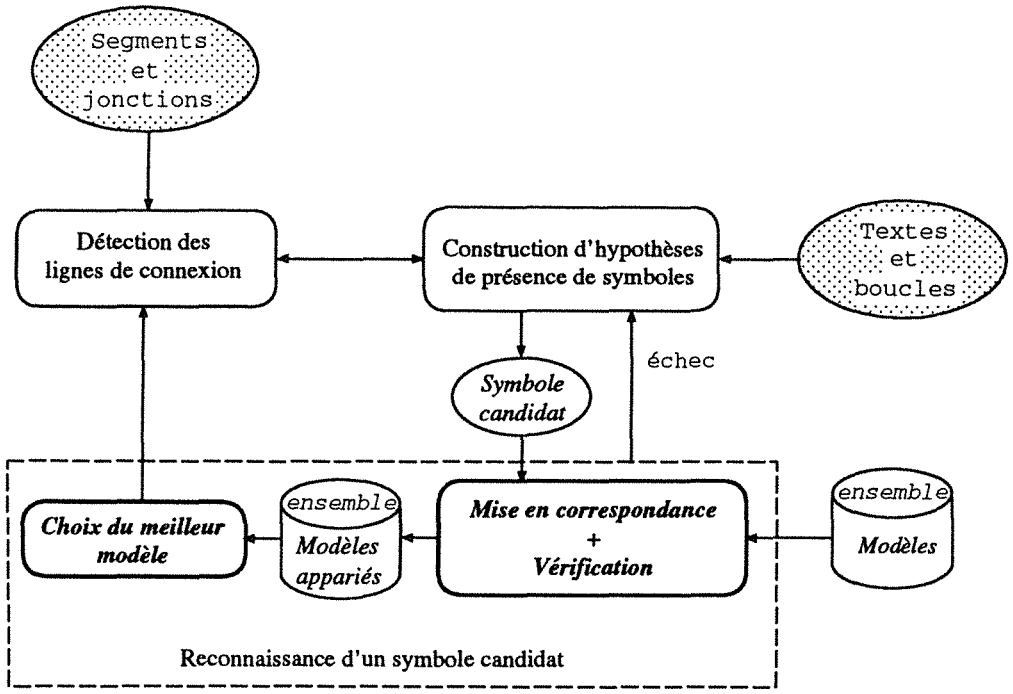


Figure 4.1. *Reconnaissance d'un symbole candidat.*

La reconnaissance est effectuée par deux modules : le premier module consiste à réaliser la mise en correspondance du candidat avec tous les modèles de base et la vérification du résultat obtenu par cet appariement. Le choix du meilleur modèle apparié avec le candidat constitue le deuxième modules.

Dans ce chapitre, nous présentons les différents modules composants le processus de reconnaissance. Les interactions de la reconnaissance avec les processus de détection des lignes de connexion et de construction d'hypothèses de présence de symboles sont détaillées dans le chapitre suivant. Le problème de mise en correspondance est présenté dans la section 4.1. La section 4.2 explicite la représentation avec laquelle nous décrivons les symboles (modèle et candidat). Le choix des caractéristiques du modèle à mettre en correspondance fait l'objet de la section 4.3. Dans la section 4.4, nous définissons nos contraintes d'appariement. Comme nous l'avons déjà vu dans le chapitre précédent, la reconnaissance doit être indépendante de la taille et de l'orientation du modèle, ce qui nous amène à prouver l'invariance de la reconnaissance par rapport à la rotation et à l'homothétie (Section 4.5). Ensuite, nous présentons notre stratégie de mise en correspondance dans la section 4.6. Enfin et avant de terminer par une discussion dans la section 4.8, l'heuristique adoptée pour le choix du meilleur modèle est exposée dans la section 4.7.

4.1 Qu'est ce qu'un problème de mise en correspondance ?

Un processus d'appariement de symboles consiste à identifier les points homologues entre le symbole candidat et le modèle. Un problème de mise en correspondance repose essentiellement sur trois points :

- *choix des primitives à apparier :* le choix des primitives à apparier dépend du problème à traiter et des primitives images qui peuvent être extraites, par exemple points, segments... Notre choix est détaillé dans la section 4.3.

- *contraintes d'appariement :* ces contraintes sont mises en œuvre pour réaliser l'appariement et pour éliminer les ambiguités. Nous avons défini trois contraintes afin de respecter la structure topologique et géométrique du modèle. Nous appelons ces contraintes par : *"contrainte de sélection de correspondants potentiels"* qui détermine l'ensemble des homologues de chacune des primitives du modèle, *"contrainte de connexité"* qui vérifie la conservation de la structure topologique du modèle dans le candidat et *"contrainte de disposition relative"* qui vérifie la préservation de la structure géométrique du modèle dans le candidat.

- *technique de mise en correspondance :* la stratégie de satisfaction de ces contraintes dépend fortement des deux premiers points. Rappelons que dans la section 2.2.4, nous avons présenté différentes techniques de mise en correspondance parmi lesquelles nous ferons notre choix.

Notons que, désirant développer une technique adaptée aux images réelles, nous avons tenu compte de la contrainte bruit inhérente aux cas réels et ce pour définir aussi bien les contraintes que la technique de mise en correspondance.

4.2 Représentation des symboles par des graphes attribués

Nous avons choisi de représenter les modèles par des graphes et plus particulièrement par des graphes *attribués* [Lee 90] [Habacha 91b]. Le fait qu'un symbole peut être considéré comme un graphe particulier où il y a de fortes contraintes topologiques et géométriques définissant sa structure et sa forme nous a guidé vers ce choix.

Les points de jonctions ou extrémités du symbole sont les sommets alors que les segments de droite ou arc de cercle sont les arêtes dans le graphe avec des attributs prenant en compte les contraintes définissant la structure du symbole. Dans la suite, le terme *point* (*segment*) sera utilisé pour désigner un point de jonction ou un point extrémité (un segment de droite ou un arc de cercle).

4.2.1 Représentation des modèles

Les primitives décrivant un symbole modèle sont les points de jonctions et les points de connexion, les segments de droite et les arcs de cercles. Rappelons qu'un *point de connexion* est un point de rattachement du symbole aux autres composantes du schéma.

Dans le modèle, les segments de droite et les arcs de cercle sont représentés par les arêtes dans le graphe avec les attributs suivants: le type du segment (segment de droite ou arc de cercle), les points extrémités et la courbure du segment. Les points sont représentés par les sommets du graphe avec les attributs suivants: les coordonnées du point, son degré, son type (point de connexion ou non), la liste des segments incidents à ce point et la liste des angles entre ces segments. Nous définissons les sommets et les arêtes du graphe représentant le symbole modèle par la définition suivante:

Définition 2 *Etant donné un graphe modèle* $(\mathcal{N}, \mathcal{A})$.

- *un sommet* $n \in \mathcal{N}$ *est défini par le quintuplé suivant:*

$$n = ((x_n, y_n); d(n); t_{point}(n); [a_n^1, \ldots, a_n^{d(n)}]; [\theta_n^1, \ldots, \theta_n^{d(n)}])$$

$$avec \begin{cases} (x_n, y_n)\ coordonnes\ du\ sommet\ n \\ d(n) = degré\ de\ n \\ t_{point}(n) = \begin{cases} 1\ si\ n\ point\ de\ connexion \\ 0\ si\ non \end{cases} \\ \forall i \in 1..d(n)\ a_n^i\ arête\ incidente\ à\ n \\ \forall i \in 1..(d(n)-1)\ \theta_n^i = Angle(a_n^i, a_n^{i+1})\ et\ \theta_n^{d(n)} = Angle(a_n^{d(n)}, a_n^1) \end{cases}$$

Notons que $\sum_{i=1}^{d(n)} \theta_n^i$ *est égale à* 2π *et que les listes* $[a_n^1, \ldots, a_n^{d(n)}]$ *et* $[\theta_n^1, \ldots, \theta_n^{d(n)}]$ *sont deux listes circulaires.*

- *une arête* $a \in \mathcal{A}$ *est défini par le triplé suivant :*

$$a = (t_{seg}(a); \{n_1, n_2\}; \rho)\ \ avec \begin{cases} t_{seg}(a) = \begin{cases} DROITE\ si\ a\ segment\ de\ droite \\ CERCLE\ si\ a\ arc\ de\ cercle \end{cases} \\ n_1, n_2\ \ sommets\ extrémités \in \mathcal{N} \\ \rho\ \ courbure \end{cases}$$

La Figure 4.2 montre un exemple de symbole modèle. Les sommets n_5 et n_6 sont les deux points de connexion de ce symbole. Le sommet n_1 et l'arête a sont respectivement définis par $n_1 = ((x_1, y_1); 2; 0; [a, b]; [\theta, 2\pi - \theta])$ et $a = (DROITE; \{n_1, n_2\}; 0)$.

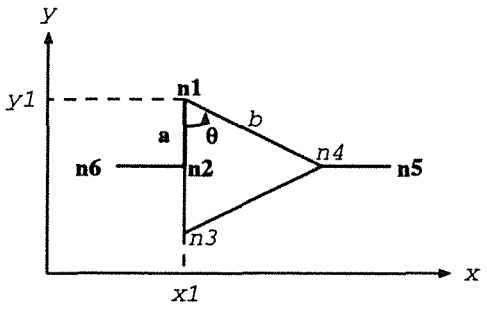


Figure 4.2. *Un amplificateur.*

Nous appelons *configuration locale* du sommet n la liste $[\theta_n^1, \ldots, \theta_n^{d(n)}]$. Nous disons que deux sommets n_1 et n_2 sont adjacents dans le graphe modèle s'il existe une arête $a \in \mathcal{A}$ dont les sommets extrémités sont n_1 et n_2.

4.2.2 Représentation des candidats

Les symboles candidats sont eux aussi représentés par des graphes attribués, puisque la mise en correspondance de deux structures identiques est plus simple à mettre en œuvre. Le graphe attribué représentant le symbole candidat est le même que celui représentant le symbole modèle. Seuls deux points sont à signaler concernant les valeurs de t_{point} (type d'un sommet du graphe) et t_{seg} (le type d'une arête) :

- avant de l'avoir reconnu, un symbole candidat n'a pas de points de connexion connus. De ce fait, à la construction du graphe, la valeur du t_{point} est égale à 0 pour tous les sommets du candidat. Une fois la mise en correspondance faite, tout sommet du graphe candidat apparié avec un sommet du modèle de type *"point de connexion"* est alors décrété comme étant un point de connexion,

- d'autre part, un graphe candidat ne contient pas d'arêtes de type *"CERCLE"*, puisqu'un arc de cercle apparaît comme une suite de segments de droite résultant de l'approximation polygonale appliquée sur cet arc de cercle. De ce fait, un arc de cercle dans le graphe modèle doit être mis en correspondance avec un ensemble de segments de droite qui l'approche dans le graphe candidat.

4.3 Choix des caractéristiques à apparier

Différentes primitives peuvent être choisies pour réaliser l'appariement comme par exemple les points, les segments, les snakes... Le choix de ces primitives dépend des problèmes à traiter et des primitives qui peuvent être extraites de l'image. La primitive segment est souvent utilisée dans les problèmes d'appariement. Citons par exemple les travaux de Ayache *et al.* [Ayache 86] pour la reconnaissance d'objets 2D ou ceux de Horaud *et al.* [Horaud 89] pour l'appariement de deux images dans le cadre de la stéréovision. Etant donné, que nous faisons de la mise en correspondance de graphes, les primitives possibles sont donc les sommets et les arêtes. Dans son approche, Kuner [Kuner 85] a mis en correspondance les sommets en utilisant les degrés comme seule information pour la recherche d'isomorphismes de graphes, afin de reconnaître les symboles dans un circuit électrique. L'algorithme qu'il a utilisé est celui d'Ullmann [Ullmann 76]. Cependant cette contrainte est trop faible pour faire de la reconnaissance de formes, puisqu'aucune information sur la structure géométrique de la forme n'est utilisée. Ainsi dans la Figure 4.3, nous pouvons voir que les deux graphes sont topologiquement isomorphes bien qu'ils ne représentent pas la même forme.

Figure 4.3. *Insuffisance de l'appariement des sommets pour la reconnaissance des formes.*

Pour notre part, nous avons choisi de mettre en correspondance deux primitives, *les sommets* et *arêtes incidentes* à ces sommets puisque le graphe est riche en information. En effet, nous proposons d'utiliser non seulement les *degrés* mais aussi les

configurations locales de ces sommets[1] contenus dans les graphes attribués. Ceci a pour avantage de respecter la structure locale du sommet à apparier.

4.4 Nos contraintes de mise en correspondance

Notons, avant de définir les différentes contraintes à utiliser pour la mise en correspondance, que le graphe candidat à apparier est généralement entaché entaché de bruits qui sont d'une part, introduits par les processus du bas niveau et qui d'autre part proviennent du module de construction d'hypothèses de présence de symbole. En effet, comme nous l'avons vu au 3.2, qui présente les étapes du bas niveau, la vectorisation de l'image peut subdiviser un segment en plusieurs petits segments ou encore perdre l'information jonction. Certains segments ou parties de segments du schéma peuvent être également perdu. En plus de ces problèmes intervenants au processus du bas niveau, notons que l'étape de construction d'hypothèses de présence de symbole ne délivre pas l'ensemble de segments parfait décrivant un symbole donné mais elle extrait une zone qui généralement contient parmi l'ensemble de segments la description d'un symbole (le graphe candidat). Ainsi, il y aura souvent des arêtes du graphe candidat qui n'appartiennent pas au symbole à reconnaître. La Figure 4.4 montre un exemple d'un circuit électrique: nous voyons un exemple de graphe candidat qui peut être extrait, il contient plusieurs segments n'appartenant pas à l'amplificateur opérationnel (symbole recherché). Sur ce schéma, nous notons l'existence de ses différentes erreurs, dont nous devons tenir compte. Pour notre part, nous classons ce bruit en deux types:

- le bruit affectant les sommets du graphe, par exemple la jonction a de la Figure 4.4 est représentée par un sommet auquel il manque un segment,
- le bruit affectant les arêtes par exemple le segment b de la même Figure 4.4 est subdivisé en plusieurs petits segments.

De ce fait, l'appariement d'un modèle avec un candidat doit être fait en prenant en considération ces erreurs. Cette prise en compte est réalisée au niveau des différentes contraintes que nous définissons dans cette section pour mettre en œuvre la mise en correspondance. Le bruit qui entache les sommets est pris en compte par la contrainte de sélection de correspondants potentiels, quant au bruit affectant les arêtes, il est traité par la contrainte de connexité.

Nous abordons à présent le problème de la formalisation et de la définition des contraintes nous permettant de mettre en correspondance deux graphes (modèle et candidat) en respectant la structure topologique et géométrique du symbole modèle.

1. Liste des angles entre les arêtes incidentes au sommet

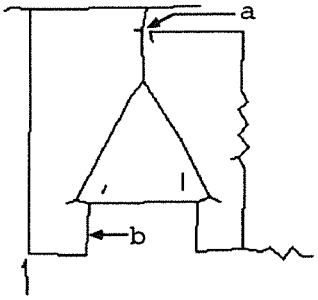


Figure 4.4. *Schéma électrique après traitement du bas niveau.*

4.4.1 Contrainte de sélection de correspondants potentiels

Dans un premier temps, il s'agit pour chaque sommet du modèle de déterminer son correspondant dans le candidat. Ceci nous amène à chercher la liste des sommets du graphe candidat constituant les correspondants potentiels. A cet effet, nous proposons une contrainte permettant de guider cette sélection.

Une approche simpliste consiste à valider une mise en correspondance d'un sommet du modèle avec un sommet du candidat si les deux sommets possèdent les mêmes configurations locales, c'est à dire s'ils possèdent les mêmes listes d'angles à une permutation circulaire près. Cependant cette condition est trop forte puisqu'elle effectue un appariement strict, exigeant que les deux sommets à mettre en correspondance doivent avoir le même degré. Ce qui sous-entend que les sommets ne peuvent pas être entachés d'erreurs, ce qui ne correspond pas à notre spécification. Par conséquent, au lieu d'exiger des configurations identiques entre chaque sommet du modèle et son correspondant dans le candidat, nous proposons de vérifier que chaque sommet du modèle est, soit une *sous-configuration*, soit une *sur-configuration* de la configuration de son correspondant potentiel. Cette contrainte permet de tenir compte des différents bruits affectant les sommets dont nous voulons tenir compte puisque :

- La condition de "sous-configuration" traduit le fait que le degré du sommet du modèle est inférieur au degré du sommet du candidat et par conséquent, il existe des arêtes dans le candidat qui sont supposées être en trop.

- La condition de "sur-configuration" traduit le fait que le degré du sommet modèle est supérieur au degré du sommet du candidat et par conséquent, il manque des arêtes au graphe candidat. Dans ce cas, les erreurs sont des *informations* manquantes.

Remarquons que notre choix pour le type de bruit traité ne couvre pas tous les

cas. Ainsi un sommet qui se verrait enlever un segment et rajouter un faux n'est pas traité. Un choix s'est imposé à nous et nous avons décidé de limiter les erreurs traités à celui d'un ajout strict ou d'un retrait strict. Le cas où les deux sommets possèdent des configurations identiques (degré du sommet du modèle est égale au degré du sommet du candidat) est pris en compte dans le premier cas (il y a zéro arêtes en trop).

Définition 3 *Toute arête supposée manquante au symbole candidat est représentée par Ω à l'appariement et sa position est déterminée en fonction de la position de son correspondant.*

La contrainte suivante formalise et traduit les conditions de sélection de correspondants potentiels que nous venons de présenter :

Contrainte 1 *Soient un sommet m du graphe modèle et un sommet c du graphe candidat définis par :*

$$\begin{cases} m &= ((x_m, y_m); d(m); t_{point}(m); [s_m^1, \ldots, s_m^{d(m)}]; [\theta_m^1, \ldots, \theta_m^{d(m)}]) \\ c &= ((x_c, y_c); d(c); 0; [s_c^1, \ldots, s_c^{d(c)}]; [\theta_c^1, \ldots, \theta_c^{d(c)}]) \end{cases}$$

Le sommet m est mis en correspondance avec c si et seulement si il existe une application d'appariement :

$$\begin{array}{rccc} Apparie : & \{s_m^i,\ i = 1..d(m)\} & \longrightarrow & \{s_c^j,\ j = 1..d(c)\} \cup \{\Omega\} \\ & s_m^i & \longmapsto & Apparie(s_m^i) \end{array}$$

telles que $\forall\ i \in\ 1..d(m)\ Angle(Apparie(s_m^i), Apparie(s_m^{i+1})) = \theta_m^i$ *et*

(i) *si* $d(m) = d(c)$: $Im(Apparie) = \{s_c^j,\ j = 1..d(c)\}$.

(ii) *si* $d(m) < d(c)$: $Im(Apparie) \subset \{s_c^j,\ j = 1..d(c)\}$.

(iii) *si* $d(m) > d(c)$: $\begin{cases} Im(Apparie) = \{s_c^j,\ j = 1..d(c)\} \cup \{\Omega\} \\ \text{il existe suffisamment d'arêtes } s_m^i \text{ tels que } Apparie(s_m^i) \neq \Omega \\ \text{(car nous n'admettons pas un taux d'erreurs trop élevé)} \end{cases}$

où $Im(Apparie)$ est l'image de l'application Apparie.

Remarque 1 *Une conséquence immédiate de cette contrainte est le fait que dans le cas (i) Apparie est une bijection sur $Im(Apparie)$, dans le cas (ii) Apparie est une injection sur $Im(Apparie)$ et enfin dans le cas (iii) Apparie est une application et la restriction de Apparie à l'ensemble $Apparie^{-1}(\{s_c^i,\ i = 1..d(c)\})$ est une bijection.*

Définition 4 *Soient m un sommet du modèle et c un sommet du candidat vérifiant la contrainte 1, nous désignons par homologue du sommet m le couple*

$$(c, [Apparie(s_m^1), \ldots, Apparie(s_m^{d(m)})])$$

et nous le notons $h = (c, [s_h^1, \ldots, s_h^{d(m)}])$ où $s_h^i = Apparie(s_m^i)$ et $[s_h^1, \ldots, s_h^{d(m)}]$ est appelé liste de correspondance d'arêtes.

Mise en application de la Contrainte 1 Nous abordons ici le calcul de la liste de correspondance d'arêtes dans chacun des trois cas de figures envisagés pour l'appariement :

- *Cas (i)* : la mise en correspondance n'est possible que si les deux sommets m et c possèdent la *même-configuration locale.* D'après la Remarque 1, il existe une bijection réalisant la correspondance entre les arêtes incidentes au sommet m et celles incidentes au sommet c. Ceci se traduit par la condition d'existence d'une bijection f définie sur les indices des angles qui permet de vérifier que la configuration locale du sommet modèle est identique à celle du sommet candidat. Cette condition est définie comme suit :

  $$\exists \textit{ une bijection } f \textit{ tel que } \forall i \in 1..deg_m\ \theta_m^i = \theta_c^{f(i)} \textit{ et } f(i+1) = f(i) + 1$$

  Dans ce cas la liste d'arêtes $[Apparie(s_m^1), \ldots, Apparies(s_m^{d(m)})]$ est égale à $[s_c^{f(1)}, \ldots, s_c^{d(m)}, s_c^1, \ldots, s_c^{f(1)-1}]$ qui est détuite par une permutation circulaire des arêtes incidentes au sommet c.

  Ce cas est illustré par la Figure 4.5. Les sommets n_1 et n sont définis par :

  $$\begin{cases} n_1 &= ((x_1, y_1); 3; 0; [a', b', c']; [\pi; \frac{\pi}{4}; \frac{3\pi}{4}]) \\ n &= ((x, y); 3; 0; [a, b, c]; [\frac{\pi}{4}; \frac{3\pi}{4}; \pi]) \end{cases}$$

  n_1 est mis en correspondance avec n et nous obtenons l'homologue $h_1 = (n, [c, a, b])$ (Les segments c, a et b sont donc respectivement mis en correspondance avec les segments a',b' et c' du modèle).

- *Cas (ii)* : La mise en correspondance n'est possible que si la configuration du sommet m du modèle est identique à une sous-configuration du sommet c du candidat. D'après la Remarque 1, il existe une injection définie de l'ensemble des arêtes incidentes à m dans celle de c. Ceci se traduit par la condition d'existence d'une injection f définie sur les indices des angles qui permet de vérifier que la configuration locale du sommet modèle est identique à une sous-configuration du sommet candidat. Cette condition est définie comme suit :

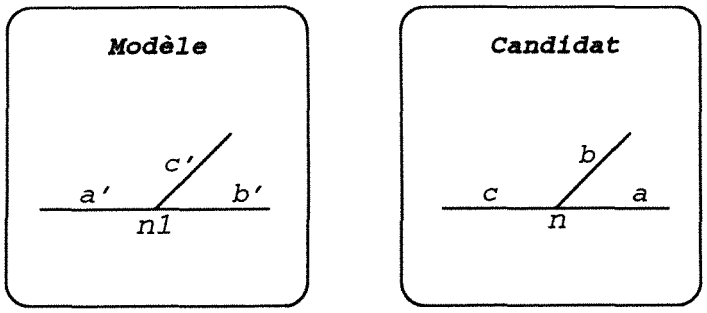


Figure 4.5. *Appariement exact.*

$$\exists \textit{ une injection} f \textit{ tel que } \forall i \in 1..deg_m\ \theta_m^i = \sum_{j=f(i)}^{f(i+1)-1} \theta_c^j \textit{ et } f(i+1) > f(i)$$

Dans ce cas la liste d'arêtes $[Apparie(s_m^1), \ldots, Appariés(s_m^{d(m)})]$ est égale à $[s_c^{f(1)}, \ldots, s_c^{f(d(m))}]$. Notons que pour tout i tel que $f(i+1) \neq f(i)+1$, $s_c^j, j = (f(i)+1)..(f(i+1)-1)$ *sont des arêtes en trop* et par conséquent, elles ne peuvent être mises en correspondance avec aucune arête incidente au sommet m du modèle.

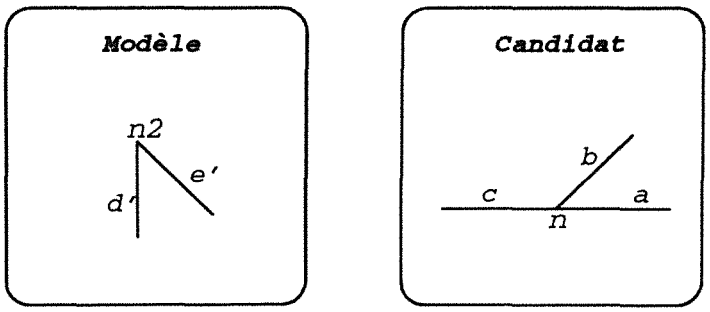


Figure 4.6. *Appariement avec une sous-configuration.*

Par exemple dans la Figure 4.6, les sommets n_2 et n sont définis par :

$$\begin{cases} n_2 &= ((x_2, y_2); 2; 0; [d', e']; [\frac{\pi}{4}; \frac{7\pi}{4}]) \\ n &= ((x, y); 3; 0; [a, b, c]; [\frac{\pi}{4}; \frac{3\pi}{4}; \pi]) \end{cases}$$

n_2 est mis en correspondance avec n et nous obtenons l'homologue $h_2 = (n, [a, b])$. Les arêtes d' et e' sont appariés respectivement avec a et b et l'arête c est supposé en trop et par conséquent, n'appartient pas au graphe candidat.

- *Cas (iii)*: c'est le troisième type d'appariement. Dans ce cas, il faut que la configuration locale du sommet c du candidat soit incluse dans celle du sommet m du modèle d'où on suppose qu'il faut ajouter les arêtes manquantes au sommet c. D'après la Remarque 1, il existe une fonction d'appariement restreinte à l'ensemble d'antécédents des arêtes incidentes au sommet c est une bijection. Ce qui se traduit par la condition d'existence d'une application f définie sur les indices des angles qui permet de vérifier que la configuration locale du sommet modèle est identique à une sur-configuration du sommet candidat. Cette condition est définie comme suit :

$$\exists \textit{ une application } f \textit{ tel que } \forall i \in 1..deg_c\ \theta_c^i = \sum_{j=f(i)}^{f(i+1)-1} \theta_m^j \textit{ et } f(i+1) > f(i)$$

Dans ce cas $\forall i = 1..d(c)$ $Apparie(s_m^{f(i)}) = s_c^i$ et pour les i tel que $f(i+1) \neq f(i)+1$, les arêtes $s_m^j, j = (f(i)+1)..(f(i+1)-1)$ correspondent à des arêtes en moins et par conséquent, sont associées à l'arête fictive Ω. Cette association indique que dans le cas d'une mise en correspondance ses arêtes doivent être ajoutées au candidat. Remarquons que c'est la même condition que dans le Cas *(ii)* en changeant les angles du modèle par ceux du candidat et réciproquement.

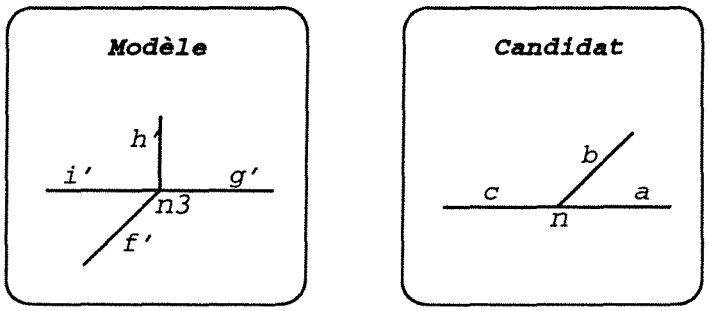


Figure 4.7. *Appariement avec une sur-configuration.*

Ceci est illustré par l'exemple donné par la Figure 4.7. Les sommets n_3 et n sont définis par :

$$\begin{cases} n_3 = ((x_3, y_3); 4; 0; [f', g', h', i']; [\frac{3\pi}{4}, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{4}]) \\ n = ((x, y); 3; 0; [a, b, c]; [\frac{\pi}{4}; \frac{3\pi}{4}; \pi]) \end{cases}$$

n_3 est mis en correspondance avec n et nous obtenons l'homologue $h_3 = (n, [b, c, \Omega, a])$. Les arêtes f', g', h' et i' sont respectivement appariés avec b, c, Ω et a ; dans ce cas il manque au sommet n une arête correspondant à h'.

Remarquons que le test d'égalité d'angles n'est pas strict ; nous tolérons une variation $\Delta\theta$ sur les angles pour tenir compte des erreurs d'imprécision des dessins à analyser.

4.4.2 Contrainte de connexité

Comme par ailleurs la connexité du modèle doit être respectée dans le candidat, il faut donc vérifier que pour tout couple de sommets du modèle (n_1, n_2) adjacents par l'arête a et appariés respectivement avec n et n', n et n' soient aussi adjacents par une arête de *même type* que l'arête a. Nous ne permettons pas d'exiger que l'adjacence de n et n' soit réalisée par exactement une arête de même type. En effet, si l'arête du modèle est de type *"CERCLE"*, les deux sommets du graphe candidat doivent être adjacents par une séquence d'arêtes qui approxime un arc de cercle. En effet, dans la Figure 4.8.a, les sommets l_1 et l_8 du candidat doivent être adjacents par un arc de cercle ; or ils sont reliés par une séquence de segments de droite résultat de l'application de l'approximation polygonale sur l'arc de cercle de l'image initiale. De même pour les arêtes du modèle de type *"DROITE"*, en général, nous n'obtenons pas un unique segment de droite mais une chaîne de segments de droite approximant le segment traité. En effet, dans la Figure 4.8.b, les sommets l_1 et l_8 du candidat doivent être adjacents par un segment de droite mais ils sont reliés par une séquence de segments de droite (c, g, d) résultat de l'approximation polygonale sur le segment c_0.

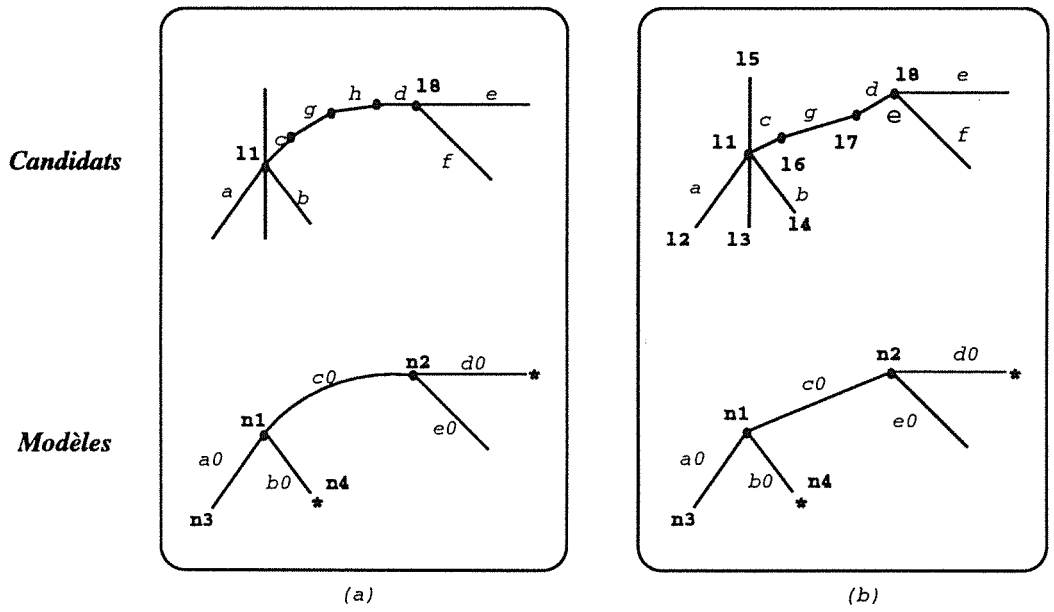


Figure 4.8. *Erreur due à l'approximation polygonale.*

Remarquons que la séquence de segments obtenue pour approximer le segment

du modèle peut lui manquer des segments ou encore peut être intersectée par un segment parasite. En effet, dans la Figure 4.9.a, la séquence de segments qui relie l_1 et l_8 est formée par (c, Ω, d) où Ω représente le segment de droite entre l_6 et l_7 supposé manquant au graphe candidat. Par ailleurs, dans la Figure 4.9.b, la séquence de segments qui relie l_1 et l_8 est formée par (c, g, d) et le segment intersectant cette séquence au sommet l_7 est supposé du bruit.

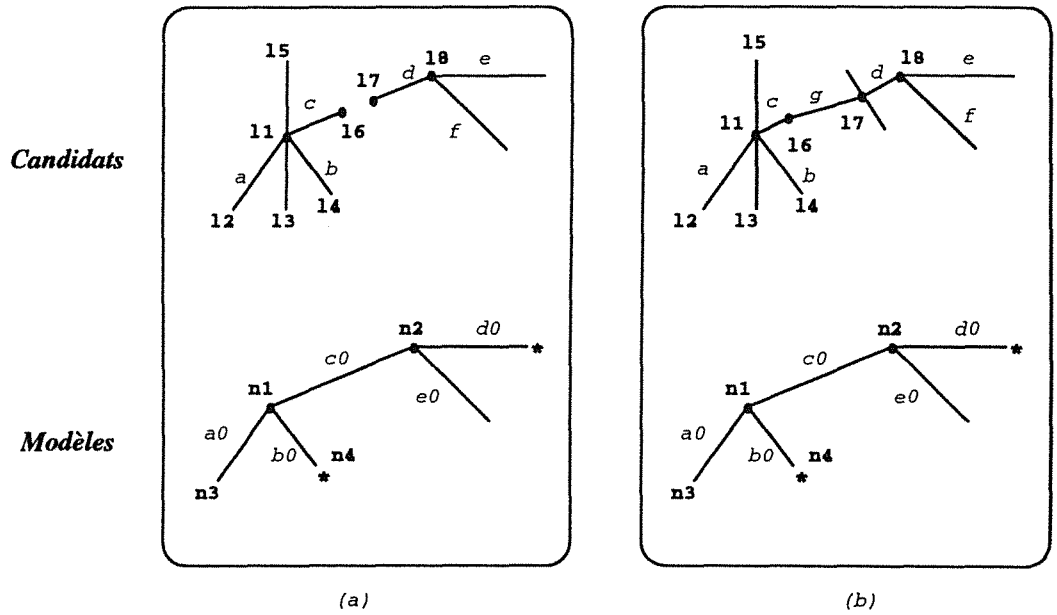


Figure 4.9. *Cas où les degrés des sommets intermédiaires de la séquence d'adjacence sont différents de deux.*

Par conséquent, la contrainte de connexité doit vérifier l'adjacence des homologues à travers une *séquence* d'arêtes de même type que celui de l'arête du modèle sans aucune contrainte sur les degrés des sommets intermédiaires de la séquence. Nous formalisons cette condition par la contrainte suivante:

Contrainte 2 *Soient n_1, n_2 deux sommets du graphe modèle mis en correspondance respectivement avec n et n' du candidat. Si n_1, n_2 sont adjacents par l'arête a alors n, n' doivent être aussi adjacents par une séquence d'arêtes qui approxime une arête de même type que a. Remarquons que cette séquence peut contenir l'arête spécial Ω.*

4.4.3 Contrainte de disposition relative

La contrainte de connexité n'est pas suffisante pour tester la compatibilité des correspondants potentiels déteminés par la Contrainte 1. En effet, comme le montre la Figure 4.10 nous avons trois exemples où pour chaque couple de graphes (modèle, candidat), la contrainte de connexité est satisfaite pour tout couple de sommets du

modèle alors que globalement le modèle et le candidat ne peuvent pas être appariés, puisque la structure géométrique du modèle n'est pas conservée. En d'autres termes, prenons par exemple les sommets n_1 et n_2 appariés respectivement par n et n', nous constatons que ou bien *la position relative* du sommet n_1 par rapport à celle de n_2 n'est pas conservée dans le candidat (Figure 4.10.a), ou bien c'est *l'orientation relative* des arêtes incidentes à n_1 par rapport à celle de n_2 qui n'est pas conservée (Figure 4.10.c). Il y a bien évidemment des cas où ni la position relative des sommets ni l'orientation relatives des arêtes n'est conservée (Figure 4.10.b).

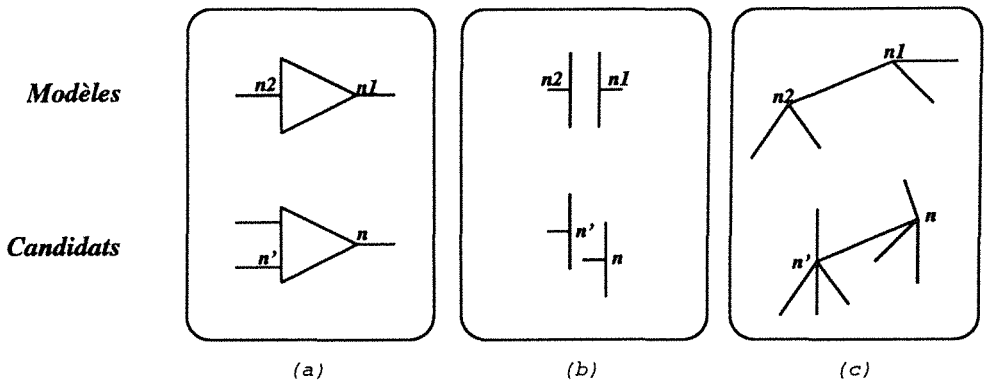


Figure 4.10. *Insuffisance de la contrainte de connexité.*

Par conséquent, il faut remédier à ce problème. Pour ce faire, nous proposons une contrainte, que nous désignons par *"Contrainte de disposition relative"*, qui assure la vérification de la conservation de la position relative des sommets et de l'orientation relative de leurs arêtes incidentes. Mais avant de présenter la contrainte premettant cette conservation, définissons tout d'abord ce qu'est la disposition relative de deux sommets d'un graphe.

A- Disposition relative de deux sommets

Etant donnés deux sommets d'un graphe quelconque, nous exprimons la disposition relative par la disposition de chacune des arêtes de leurs deux listes d'arêtes incidentes par rapport au vecteur qui relie ces deux sommets. Ceci prend en compte en même temps l'information de position relative des sommets et l'information d'orientation relative des arêtes incidentes aux sommets. Ainsi, nous définissons la disposition relative de deux sommets comme étant les deux ensembles d'angles définis entre le vecteur reliant ces deux sommets et leurs arêtes incidentes. La définition suivante donne la disposition relative de deux sommets quelconque d'un graphe :

Définition 5 *Soient deux sommets n_1 et n_2 :*

$$\begin{cases} n_1 = ((x_{n_1}, y_{n_1}); d(n_1); t_{point}(n_1); [s^1_{n_1}, \ldots, s^{d(n_1)}_{n_1}]; [\theta^1_{n_1}, \ldots, \theta^{d(n_1)}_{n_1}]) \\ n_2 = ((x_{n_2}, y_{n_2}); d(n_2); t_{point}(n_2); [s^1_{n_2}, \ldots, s^{d(n_2)}_{n_2}]; [\theta^1_{n_2}, \ldots, \theta^{d(n_2)}_{n_2}]) \end{cases}$$

La disposition relative des deux sommets n_1 et n_2 est définie par le couple $(A^{n_2}_{n_1}, A^{n_1}_{n_2})$

$$\text{où} \begin{cases} A^{n_2}_{n_1} = \{Angle(\overrightarrow{n_1 n_2}, s^i_{n_1}),\ i = 1..d(n_1)\} \\ A^{n_1}_{n_2} = \{Angle(\overrightarrow{n_1 n_2}, s^i_{n_2}),\ i = 1..d(n_2)\} \end{cases}$$

La Figure 4.11 montre un exemple qui illustre cette définition. Notons que pour le calcul des angles $Angle(\overrightarrow{n_1 n_2}, s^i_{n_j})$, les arêtes $s^i_{n_j}$ seront orientées de façon à ce que le sommet n_j soit l'origine.

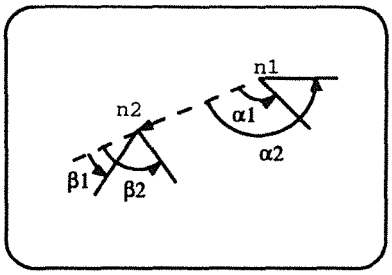


Figure 4.11. *La disposition relative de n_1 et n_2 : $(A^{n_2}_{n_1}, A^{n_1}_{n_2}) = (\{\alpha_1, \alpha_2\}, \{\beta_1, \beta_2\})$.*

B- Conservation de la disposition relative

Etant données deux couples de sommets quelconques (n_1,n_2) et (n_3,n_4), afin que ces deux couples aient des dispositions relatives identiques il faut et il suffit que l'une des deux conditions suivantes soit satisfaite :

$$(A^{n_2}_{n_1}, A^{n_1}_{n_2}) = (A^{n_4}_{n_3}, A^{n_3}_{n_4}) \text{ ou } (A^{n_2}_{n_1}, A^{n_1}_{n_2}) = (A^{n_3}_{n_4}, A^{n_4}_{n_3}) \quad (4.1)$$

La Figure 4.12 montre un exemple de deux couples de sommets n'ayant pas la même disposition relative. En effet, $(A^{n_2}_{n_1}, A^{n_1}_{n_2}) = (\{\alpha_1, \alpha_2\}, \{\beta_1, \beta_2\})$ et $(A^{n_4}_{n_3}, A^{n_3}_{n_4}) = (\{\alpha_3, \alpha_4\}, \{\beta_3, \beta_4\})$. Or $\alpha_2 \neq \alpha_3$ et par conséquent, l'équation 4.1 n'est pas satisfaite.

Comme nous l'avons souligné, nous cherchons donc à garantir que deux sommets du modèle aient la même disposition relative que leurs *homologues respectifs* qui répondent à la Contrainte 1. En fait, nous voulons qu'étant donnés deux sommets n_1 et n_2 du modèle appariés aux sommets n et n' du candidat tels que h_1 et h_2 leurs homologues respectifs, n_1 et n_2 aient la même disposition relative que h_1 et h_2. Pour cela nous définissons la disposition relative de deux homologues par analogie de celle de deux sommets définie de la façon suivante :

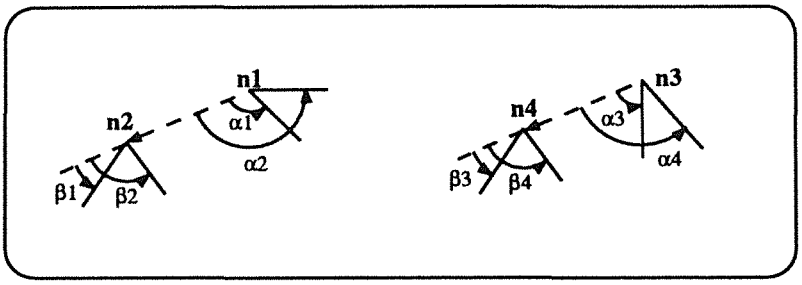


Figure 4.12. *Deux couples de sommets n'ayant pas la même disposition relative.*

Définition 6 *Soient deux sommets n_1 et n_2 du modèle appariés respectivement à n et n' du candidat tels que h_1 et h_2 soient deux homologues respectifs :*

$$\begin{cases} h_1 = (n, [s^1_{h_1}, \ldots, s^{d(n_1)}_{h_1}]) \\ h_2 = (n', [s^1_{h_2}, \ldots, s^{d(n_2)}_{h_2}]) \end{cases}$$

La disposition relative des deux homologues h_1 et h_2 est définie par le couple $(A^{h_2}_{h_1}, A^{h_1}_{h_2})$

$$où \begin{cases} A^{h_2}_{h_1} = \{Angle(\overrightarrow{nn'}, s^i_{h_1}),\ i = 1..d(n_1)\} \\ A^{h_1}_{h_2} = \{Angle(\overrightarrow{nn'}, s^i_{h_2}),\ i = 1..d(n_2)\} \end{cases}$$

En fait au lieu de vérifier la disposition relative entre sommets homologues en utilisant l'équation 4.1 il suffit de montrer que :

$$\begin{cases} Angle(\overrightarrow{n_1n_2}, s^{i_0}_{n_1}) = Angle(\overrightarrow{nn'}, s^{i_0}_{h_1}) \text{ où } i_0 \in\ 1..d(n_1) & (1) \\ Angle(\overrightarrow{n_1n_2}, s^{j_0}_{n_2}) = Angle(\overrightarrow{nn'}, s^{j_0}_{h_2}) \text{ où } j_0 \in\ 1..d(n_2) & (2) \end{cases}$$

Démontrons notre assertion. En effet, soit $i_0 \in\ 1..d(n_1)$ et supposons que $Angle(\overrightarrow{n_1n_2}, s^{i_0}_{n_1}) = Angle(\overrightarrow{nn'}, s^{i_0}_{h_1})$, comme par définition du sommet n_1 :

$$\forall i \in\ 1..d(n_1)\ Angle(\overrightarrow{n_1n_2}, s^i_{n_1}) = Angle(\overrightarrow{n_1n_2}, s^{i_0}_{n_1}) + \sum_{k=i_0}^{i-1} \theta^k_{n_1}$$

nous avons donc en utilisant *(1)*

$$Angle(\overrightarrow{n_1n_2}, s^i_{n_1}) = Angle(\overrightarrow{nn'}, s^{i_0}_{h_1}) + \sum_{k=i_0}^{i-1} \theta^k_{n_1}$$

Or h_1 est un homologue de n_1 et par conséquent,

$$Angle(\overrightarrow{nn'}, s^{i_0}_{h_1}) + \sum_{k=i_0}^{i-1} \theta^k_{n_1} = Angle(\overrightarrow{nn'}, s^i_{h_1})$$

nous obtenons donc

$$Angle(\overrightarrow{n_1 n_2}, s^i_{n_1}) = Angle(\overrightarrow{nn'}, s^i_{h_1}) \; \forall i \in 1..d(n_1) \quad \diamond$$

Nous venons donc de démontrer que si l'égalité (1) est vérifiée pour une valeur quelconque de i_0, nous avons $A^{n_2}_{n_1} = A^{h_2}_{h_1}$. Une démarche analogue permet de montrer que si l'égalité (2) est vérifiée alors $A^{n_1}_{n_2} = A^{h_1}_{h_2}$; et voie de par conséquence l'équation 4.1 est vérifiée.

Formalisons à présent notre résultat concernant la contrainte de disposition relative et commençons par définir son domaine d'application en répondant à la question suivante : *"quels sont les couples de sommets du modèle dont nous devons préserver la disposition relative dans le candidat ?"*

Pour répondre à cette question, remarquons déjà que la position d'un point mis en correspondance avec un point de connexion du modèle varie et dépend de l'étape de sélection de la zone candidate. En effet, dans un schéma, la notion de point de connexion est difficile à formaliser puisqu'il n'y a pas d'unicité du point de connexion : théoriquement tous les points appartenants à la ligne de connexion peuvent être considérés comme points de connexion. La Figure 4.13 montre trois points de connexion possibles (intersection des lignes pointillées avec la ligne de connexion L) dépendants de la taille de la zone sélectionnée.

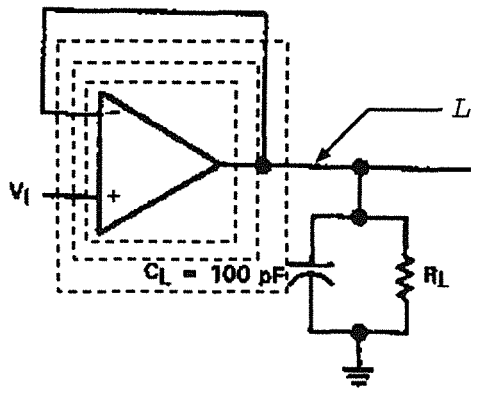


Figure 4.13. *Variation de la position du point correspondant à un point de connexion.*

Par conséquent, la position relative d'un point de connexion par rapport à tous les autres points (sauf celui qui lui est adjacent) varie en fonction du graphe candidat

sélectionné et ne peut être préservée. Ce qui explique le fait que pour ces points la notion de disposition relative n'est pas pertinente. En effet, dans la Figure 4.14, la disposition relative de n_1 et n_2 n'est pas identique à celle de n et n' puisque $\alpha_1 \neq \alpha_2$.

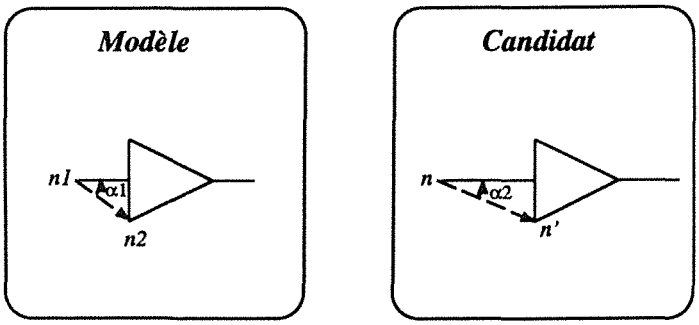


Figure 4.14. *La non conservation de disposition relative des points de connexion.*

Cependant, Bien que la disposition relative des points de connexions ne soit pas conservée, le modèle et le candidat doivent être mis en correspondance. Pour ces raisons, la disposition relative n'est pas préservée pour les couples de sommets non adjacents dont l'un des sommets est un point de connexion. La condition permettant la conservation de la disposition relative de couples d'homologues est alors définie par :

Contrainte 3 *Soient n_1 et n_2 deux sommets du modèle définis par:*

$$\begin{cases} n_1 = ((x_{n_1}, y_{n_1}); d(n_1); t_{point}(n_1); [s^1_{n_1}, \ldots, s^{d(n_1)}_{n_1}]; [\ldots]) \\ n_2 = ((x_{n_2}, y_{n_2}); d(n_2); t_{point}(n_2); [s^1_{n_2}, \ldots, s^{d(n_2)}_{n_2}]; [\ldots]) \end{cases}$$

tels que si n_1, n_2 non adjacents alors $t_{point}(n_1) \neq 1$ et $t_{point}(n_2) \neq 1$. Si n_1 et n_2 sont appariés avec n et n' tels h_1 et h_2 leurs homologues respectifs définis par:

$$\begin{cases} h_1 = (n, [s^1_{h_1}, \ldots, s^{d(n_1)}_{h_1}]) \\ h_2 = (n', [s^1_{h_2}, \ldots, s^{d(n_2)}_{h_2}]) \end{cases}$$

alors h_1 et h_2 possèdent la même disposition relative que n_1 et n_2 si et seulement si

$$\begin{cases} Angle(\overrightarrow{n_1 n_2}, s^{i_0}_{n_1}) = Angle(\overrightarrow{nn'}, s^{i_0}_{h_1}) \text{ où } i_0 \in 1..d(n_1) \\ Angle(\overrightarrow{n_1 n_2}, s^{j_0}_{n_2}) = Angle(\overrightarrow{nn'}, s^{j_0}_{h_2}) \text{ où } j_0 \in 1..d(n_2) \end{cases}$$

Remarque 2 *Le vecteur $\overrightarrow{n_1 n_2}$ a pour correspondant le vecteur $\overrightarrow{nn'}$.*

La Figure 4.15 illustre le fait que la Contrainte 3 n'est pas satisfaite pour les trois exemples présentés à la Figure 4.10. En effet, n_1 et n_2 sont respectivement appariés avec n et n'. D'après la Contrainte 3, il faut vérifier que $Angle(\overrightarrow{n_1n_2}, s^1_{n_1}) = Angle(\overrightarrow{nn'}, s^1_{h_1})$ et $Angle(\overrightarrow{n_1n_2}, s^1_{n_2}) = Angle(\overrightarrow{nn'}, s^1_{h_2})$ (nous faisons la vérification pour les valeurs $i_0 = 1$ et $j_0 = 1$). Dans les trois cas *(a)*, *(b)* et *(c)*, nous avons $Angle(\overrightarrow{n_1n_2}, s^1_{n_1}) = \alpha_1$ et $Angle(\overrightarrow{nn'}, s^1_{h_1}) = \beta$ or $\beta \neq \alpha_1$; la première condition exigée par la Contrainte 3 n'est pas vérifiée, ce qui implique que cet appariement ne conserve pas la disposition relative des sommets n_1 et n_2.

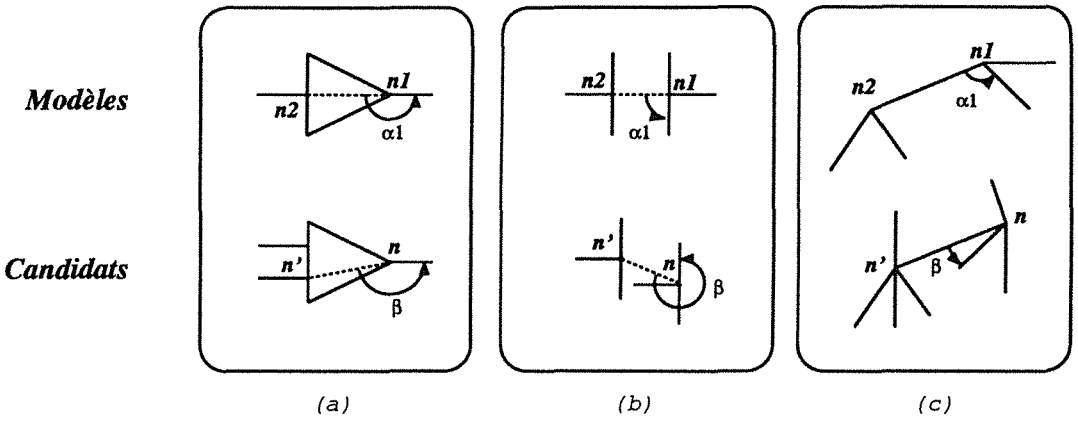


Figure 4.15. *La non conservation de la disposition relative des sommets n_1 et n_2.*

C- Utilisation de la cohérence de la correspondance d'arêtes dans la contrainte de disposition relative

Dans le cas où les deux sommets du modèle sont adjacents, la Contrainte 3 peut être respectée de façon plus simple et moins coûteuse. Nous allons expliquer comment, au lieu de vérifier la conservation de la disposition relative par l'égalité des angles pour les sommets adjcents, il suffit de tester *la cohérence de la correspondance d'arêtes* dans les homologues.

En regardant de plus près la Contrainte 3, nous constatons que dans le cas où les sommets sont adjacents nous n'avons pas besoin de calculer le couple d'angles pour vérifier la conservation de la disposition relative. En effet, si les deux sommets n_1 et n_2 sont adjacents par une arête a, nous pouvons l'utiliser comme lien entre ces deux sommets à la place du vecteur $\overrightarrow{n_1n_2}$. D'après la Remarque 2, il suffit donc de tester que le correspondant de l'arête a lie les deux sommets n et n' pour satisfaire le fait que le vecteur $\overrightarrow{n_1n_2}$ soit mis en correspondance avec $\overrightarrow{nn'}$. Or l'arête a a été appariée séparément deux fois : une fois à la mise en correspondance de n_1 avec n et une deuxième fois au moment de l'appariement de n_2 par n'. Notons par a_1 et a_2 ces

deux arêtes. Par conséquent, nous devons vérifier la cohérence de correspondance d'arêtes c'est à dire que a_1 et a_2 sont les mêmes. La Figure 4.16 montre comment cette cohérence n'est pas vérifiée pour l'exemple de la Figure 4.15.c où les deux sommets n_1 et n_2 sont adjacents. n_1 (resp. n_2) est apparié avec n (resp. n'). Soit h_1 (resp. h_2) homologue de n_1 (resp. n_2). Nous remarquons que c_0 correspond à d dans h_1 et à c dans h_2, avec $c \neq d$.

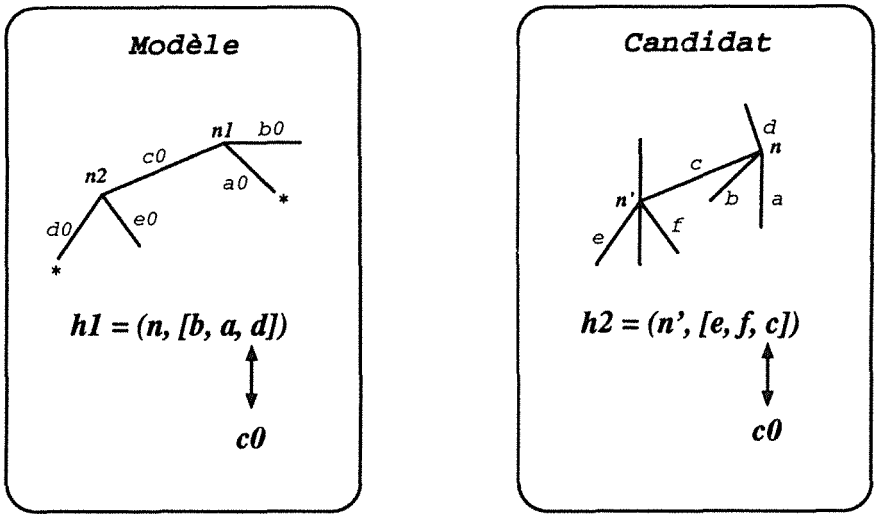


Figure 4.16. *La non cohérence de la correspondance d'arêtes.*

Remarquons que ce test de cohérence ne doit pas être strict, c'est à dire que nous ne devons pas tester l'égalité des deux arêtes mais leur appartenance à une même chaîne et ceci pour tenir compte du bruit. En effet, dans la Figure 4.17, le segment c_0 correspond à c dans l'homologue $h_1 = (l_1, [b, c, a])$ de n_1 et à d dans l'homologue $h_2 = (l_8, [e, d, f])$ de n_2. Bien que c soit différent de d, la disposition relative de n_1 et n_2 est conservée. Dans ce cas la cohérence de la correspondance d'arêtes doit être considérée comme vérifiée et par conséquent, au lieu de tester que $c = d$ il faut donc vérifier que les arêtes c et d constituent le début et la fin d'une séquence d'arêtes reliant les sommets l_1 et l_8.

Nous avons donc reconsidéré, à la lumière de cette étude plus approfondie du cas où les sommets sont adjacents, la définition de la Contrainte 3 afin de la définir par cas. Par conséquent, suivant l'adjacence ou pas des sommets sont , nous vérifions la cohérence de correspondance d'arêtes ou la conservation d'angles. La contrainte suivante traduit ces conditions :

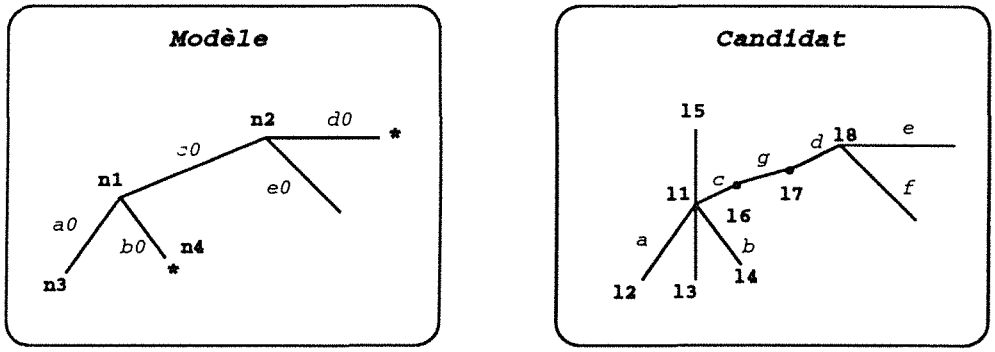


Figure 4.17. *Prise en compte du bruit dans le test de cohérence de correspondance d'arêtes.*

Contrainte 3' *Soient n_1 et n_2 deux sommets du modèle définis par:*

$$\begin{cases} n_1 = ((x_{n_1}, y_{n_1}); d(n_1); t_{point}(n_1); [s^1_{n_1}, \ldots, s^{d(n_1)}_{n_1}]; [\ldots]) \\ n_2 = ((x_{n_2}, y_{n_2}); d(n_2); t_{point}(n_2); [s^1_{n_2}, \ldots, s^{d(n_2)}_{n_2}]; [\ldots]) \end{cases}$$

et appariés avec n et n' tels h_1 et h_2 leurs homologues respectifs définis par:

$$\begin{cases} h_1 = (n, [s^1_{h_1}, \ldots, s^p_{h_1}, \ldots, s^{d(n_1)}_{h_1}]) \\ h_2 = (n', [s^1_{h_2}, \ldots, s^q_{h_2}, \ldots, s^{d(n_2)}_{h_2}]) \end{cases}$$

n_1 et n_2 possèdent la même disposition relative que h_1 et h_2 si et seulement si:

(i) Si n_1, n_2 sont adjacents par l'arête a mise en correspondance avec $s^p_{h_1}$ dans h_1 et $s^q_{h_2}$ dans h_2, alors n et n' doivent l'être aussi par une séquence d'arêtes $(c_1, \ldots, c_n)$ telle que $(c_1 = s^p_{h_1})$, $(c_n = s^q_{h_2})$ et Ω peut appartenir à cette séquence.

(ii) Si n_1, n_2 sont non adjacents tels que $t_{point}(n_1) \neq 1$ et $t_{point}(n_2) \neq 1$ alors:

$$\begin{cases} Angle(\overrightarrow{n_1 n_2}, s^{i_0}_{n_1}) = Angle(\overrightarrow{nn'}, s^{i_0}_{h_1}) \text{ où} i_0 \in 1 \ldots d(n_1) \\ Angle(\overrightarrow{n_1 n_2}, s^{j_0}_{n_2}) = Angle(\overrightarrow{nn'}, s^{j_0}_{h_2}) \text{ où} j_0 \in 1 \ldots d(n_2) \end{cases}$$

4.4.4 Formulation de notre problème d'appariement

En résumé soit $(\mathcal{N}, \mathcal{A})$ le graphe attribué représentant le modèle avec $|\mathcal{N}| = n$ et $|\mathcal{A}| = a$, notre problème de mise en correspondance peut être vu comme un problème de propagation de contraintes avec l'énoncé suivant :

- $\mathcal{V} = \mathcal{N}$: ensemble des sommets du modèle avec $|\mathcal{V}| = v$.

- $\mathcal{E} = \{(n_i, n_j) \ / \ n_i, n_j \in \mathcal{V}$ *et* $t_{point}(n_i) \neq 1$ *et* $t_{point}(n_j) \neq 1$ *si* $\{n_i, n_j\} \notin \mathcal{A}\}$: ensemble des arcs crées par nos contraintes avec $|\mathcal{E}| = e$.

- $\mathcal{H}_i = \{h_b = (n, [s^1_{h_1}, \ldots, s^{d(n_i)}_{h_1}])\}$: ensemble des homologues du sommet n_i.

Le problème consiste alors à calculer les ensembles $\mathcal{H}_i$ en utilisant la Contrainte 1 de sélection de correspondants potentiels (§ 4.4.1) et à trouver un n-uplet appartenant à $\mathcal{H}_1 \times \ldots \times \mathcal{H}_n$ qui satisfait la Contrainte 2 de connexité (§ 4.4.2) et la Contrainte 3' de disposition relative (§ 4.4.3).

La Figure 4.18.b montre deux exemples de graphes $(\mathcal{V}, \mathcal{E})$ construits à partir des deux graphes modèles représentant une capacité et un amplificateur donnés par la Figure 4.18.a. Aux graphes modèles, nous ajoutons les arcs crées par la contrainte de disposition relative. Ces arcs relient tous les sommets non adjacents du modèle sauf les sommets du type point de connexion puisqu'ils n'interviennent pas dans cette contrainte. Dans la Figure 4.18, ces sommets sont marqués par une croix. Remarquons que lors de l'implantation, ces arcs ne sont crées physiquement.

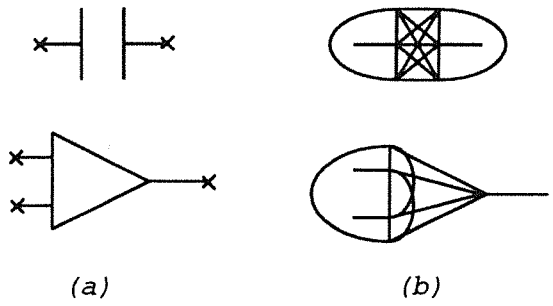


Figure 4.18. *Exemples de graphes* $(\mathcal{V}, \mathcal{E})$.

Pour résoudre ce problème d'appariement, il resterait à définir une stratégie nous permettant de satisfaire ces contraintes. Néanmoins, il ne parait pas possible de passer sous silence la propriété d'invariance de la reconnaissance par rapport à la rotation et l'homothétie. En effet, rappelons que l'un de nos objectifs consiste à reconnaître les symboles indépendamment de leur *orientation* et de leur *taille* (§ 3.1.3). Pour cette raison avant de traiter le problème de stratégie, il nous a paru nécessaire d'examiner en détail ce point en prouvant cette propriété d'invariance de la reconnaissance.

4.5 Invariance de la reconnaissance par rapport à la rotation et l'homothétie

Prouver que la reconnaissance est indépendante de l'orientation et de la taille des symboles, nous conduit à prouver que si un graphe candidat C est apparié avec un graphe modèle M alors l'image de ce candidat par l'une des deux transformations, *rotation ou homothétie*, est forcément appariée avec le même modèle M. Ce qui revient à démontrer l'équation 4.2 :

$$si\ \mathcal{F}(C) = M\ alors\ \mathcal{F}(\mathcal{T}(C)) = M \tag{4.2}$$

où $\mathcal{F}$ est la fonction de mise en correspondance et $\mathcal{T}$ est l'une des transformations : rotation, homothétie ou la composition des deux. Pour cela il suffit de prouver que toutes les contraintes que nous venons de définir vérifient cette équation.

Avant d'examiner ces contraintes une par une, nous tenons à rappeler des propriétés évidentes de la géométrie affine. C'est dans ces propriétés où réside l'explication de la preuve de l'équation 4.2 pour chacune des contraintes.

Propriété 1 *Les transformations affines conservent les angles.*

Propriété 2 *L'image d'une arête du graphe par une transformation affine est l'arête reliant l'image des deux sommets extrémités.*

La deuxième propriété exprime le fait que l'information topologique n'est pas affectée par ces transformations géométriques. Notons aussi que la rotation et l'homothétie sont des transformations affines et par conséquent, vérifient les deux propriétés précédentes.

- **Invariance de la contrainte de sélection de correspondants potentiels :** nous voulons prouver que si un sommet n_1 du graphe modèle M est mis en correspondance avec le sommet n du graphe candidat C par la contrainte de sélection de correspondants potentiels (Contrainte 1) alors n_1 est apparié avec $\mathcal{T}(n)$. Il suffit donc de vérifier que les sommets n et $\mathcal{T}(n)$ ont la même configuration locale. Soit un sommet n du candidat défini par :

$$n = ((x_n, y_n); d(n); t_{point}(n); [s_n^1, \ldots, s_n^{d(n)}]; [\theta_n^1, \ldots, \theta_n d(n)])$$

Posons $\mathcal{T}(n) = n'$ avec n' défini par :

$$n' = ((x_{n'}, y_{n'}); d(n'); t_{point}(n'); [s_{n'}^1, \ldots, s_{n'}^{d(n')}]; [\theta_{n'}^1, \ldots, \theta_{n'}^{d(n')}])$$

Nous avons $d(\mathcal{T}(n)) = d(n)$ d'après la Propriété 2. Nous avons par définition $\theta_n^i = Angle(s_n^i, s_n^{i+1})$ or $Angle(s_n^i, s_n^{i+1}) = Angle(\mathcal{T}(s_n^i), \mathcal{T}(s_n^{i+1}))$ d'après la Propriété 1 et par conséquent,

$$\mathcal{T}(n) = ((x_{n'}, y_{n'}); d(n); t_{point}(n); [\mathcal{T}(s_n^1), \ldots, \mathcal{T}(s_n^{d(n)})]; [\theta_n^1, \ldots, \theta_n^{d(n)}])$$

En conclusion n et $\mathcal{T}(n)$ possèdent la même configuration, et donc si n est un correspondant potentiel de n_1 alors $\mathcal{T}(n)$ l'est aussi.

- **Invariance de la contrainte de connexité :** évident d'après la Propriété 2.
- **Invariance de la contrainte de disposition relative :** nous voulons prouver que si deux sommets n_1 et n_2 du graphe modèle M sont mis respectivement en correspondance avec les sommets n et n' du graphe candidat C et satisfont la Contrainte 3 de disposition relative alors n_1 et n_2 appariés respectivement avec $\mathcal{T}(n)$ et $\mathcal{T}(n')$ satisferont cette Contrainte 3 (Figure 4.19). Nous voulons montrer pour $i_0 \in 1 \ldots d(n_1)$ et $j_0 \in 1 \ldots d(n_2)$ l'implication suivante :

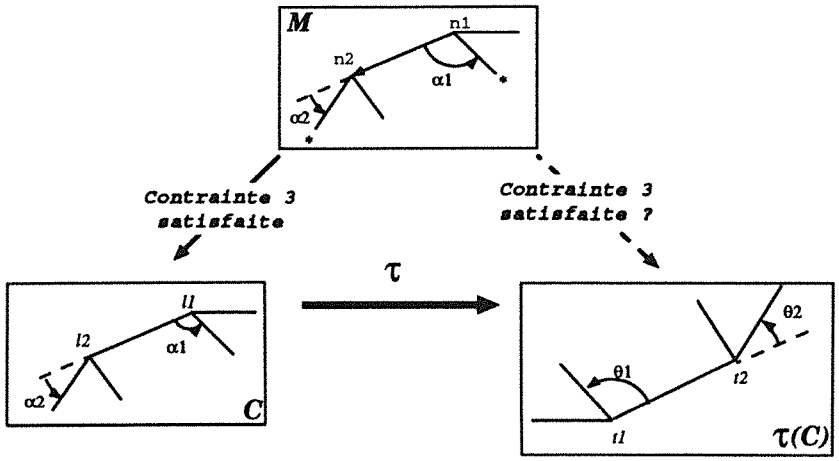


Figure 4.19. *Invariance de la contrainte de disposition relative : M et $\mathcal{T}(C)$ satisfont la Contrainte 3, si nous prouvons que $\theta_1 = \alpha_1$ et $\theta_2 = \alpha_2$.*

$$\begin{array}{l} si\ on\ a \left\{ \begin{array}{l} Angle(\overrightarrow{nn'}, s_{h_1}^{i_0}) = Angle(\overrightarrow{n_1 n_2}, s_{n_1}^{i_0}) \\ Angle(\overrightarrow{nn'}, s_{h_2}^{j_0}) = Angle(\overrightarrow{n_1 n_2}, s_{n_2}^{j_0}) \end{array} \right. \\ alors \left\{ \begin{array}{l} Angle(\overrightarrow{\mathcal{T}(n)\mathcal{T}(n')}, \mathcal{T}(s_{h_1}^{i_0})) = Angle(\overrightarrow{n_1 n_2}, s_{n_1}^{i_0}) \\ Angle(\overrightarrow{\mathcal{T}(n)\mathcal{T}(n')}, \mathcal{T}(s_{h_2}^{j_0})) = Angle(\overrightarrow{n_1 n_2}, s_{n_2}^{j_0}) \end{array} \right. \end{array} \quad (4.3)$$

D'après la Propriété 1, nous avons :

$$\begin{cases} Angle(\overrightarrow{\mathcal{T}(n)\mathcal{T}(n')}, \mathcal{T}(s^{i_0}_{h_1})) = Angle(\overrightarrow{nn'}, s^{i_0}_{h_1}) \\ Angle(\overrightarrow{\mathcal{T}(n)\mathcal{T}(n')}, \mathcal{T}(s^{j_0}_{h_2})) = Angle(\overrightarrow{nn'}, s^{j_0}_{h_2}) \end{cases} \quad (4.4)$$

Et par conséquent, l'équation 4.3 est satisfaite. Ainsi nous avons prouvé l'invariance de la Contrainte 3 par rapport aux deux transforamtions affines (rotation et homothétie).

En résumé, nous avons réalisé le choix des primitives à apparier et défini nos contraintes d'appariement qui permettent de respecter la structure topologique et géométrique du modèle de manière indépendante de l'orientation et de la taille du modèle. Pour résoudre le problème de mise en correspondance, il ne reste qu'à définir la stratégie de satisfaction de ces contraintes.

4.6 Notre stratégie d'appariement

Cette section est consacrée au problème de satisfaction des contraintes définies au § 4.4. Avant de définir notre stratégie d'appariement, nous étudions d'abord l'applicabilité des différentes techniques de mise en correspondance présentées dans § 2.2.4.

4.6.1 A propos de l'applicabilité des techniques d'appariement à notre problème

Un examen des différentes techniques de mise en correspondance présentées dans § 2.2.4 est réalisé afin de pouvoir définir celle qui s'adapte au mieux à notre problème.

Transformée de Hough : cette technique peut être appliquée à notre problème puisque nous pouvons considérer ce dernier comme un problème paramétrable. En effet, une paramétrisation possible consiste à estimer l'homothétie et la rotation qui existent entre le symbole modèle et le symbole candidat. L'utilisation de cette approche est présentée dans [Habacha 91b]. Cependant, c'est pour deux raisons différentes que nous avons jugé que cette approche ne répond pas au mieux à nos objectifs :

- les valeurs des paramètres de l'homothétie et de la rotation ne sont pas utiles en elles mêmes pour l'analyse du document complet. En effet, un même symbole peut apparaître dans le schéma avec deux angles de rotations différents par rapport au modèles. En plus de ceci supposer que pour un schéma donné, le rapport d'homothétie est le même pour tous les symboles nécessite que la base de modèles soit normalisée ; ceci constitue une restriction non nécessaire pour

la partie analyse. Par conséquent, l'utilisation d'une telle technique nécessite de déterminer l'image des points de connexions par la transformation trouvée,

- la deuxième raison qui est plus importante consiste à une constation purement expérimentale. En effet, nous avons constaté que nous n'obtenons pas de *"vrai"* points d'accumulations à cause du nombre de données qui est assez faible; les votes sont éparpillés sur les accumulateurs à cause du fait qu'un symbole modèle est constitué au maximum d'une dizaine de sommets.

Prédiction-vérification : cette technique peut être appliquée pour estimer la transformation qui existe entre le symbole modèle et le symbole candidat. Mais comme nous l'avons dit dans l'approche précédente nous n'avons pas besoin de connaître les valeurs du rapport d'homothétie ou de l'angle de rotation. En plus de ceci rappelons que l'un des problèmes majeurs de la prédiction-vérification réside dans la détermination du premier choix d'appariement qui doit être suffisament robuste pour ne pas être faux. Or nous ne possédons aucun critère que nous jugeons assez robuste pour réaliser ce premier choix. Pour toutes ces raisons, il nous a paru plus sage de ne pas utiliser cette technique.

Clique maximale : dans notre cas, le problème de la fermeture transitive (cf 2.2.4) ne se pose pas puisque pour tout couple d'hypothèse, déterminé par la contrainte de sélection de correspondants potentiels, nous pouvons calculer la relation de compatibilité, en utilisant les contraintes de connexité et de disposition relative déjà définies. C'est une solution très coûteuse si le nombre de sommets du graphes de compatibilités est élevé du fait du comportement exponentiel de la recherche de la plus grande clique maximale.

Relaxation discrète et relaxation continue : Notre problème se présente comme un problème de propagation de contraintes *binaires à valeurs discrètes.* La technique de relaxation discrète est donc applicable sans aucune modification dans la formulation du problème et en particulier l'algorithme *AC4*. Mais comme nous l'avons déjà mentionné l'utilisation d'un tel algorithme nécessite une étape de vérification de la consistance globale de la solution. Remarquons que le relachement des contraintes a été fait puisque les contraintes de connexité et de disposition relative tiennent compte des erreurs qui peuvent affecter le symbole candidat. Par contre la technique de relaxation continue ne peut pas être appliquée directement à notre problème comme il a été formulé puisque les contraintes définies sont à valeurs discrètes. Il nous faut donc transformer nos contraintes et définir différentes *propabilités associées aux homologues* ainsi que des fonctions de compatibilité à *valeurs continues* garantissant la *convergence* du processus.

4.6.2 Notre approche hiérarchique

Pour éviter l'explosion combinatoire, nous avons opté pour une stratégie hiérarchique qui s'exécute en deux étapes: la première étape élimine les homologues inconsistant localement; puis la deuxième étape teste la consistance globale sur un ensemble réduit d'homologues. Afin d'implanter l'étape d'élimination des étiquettes inconsistantes, nous avons choisi une technique de relaxation discrète mise en œuvre par l'algorithme *AC4*. Rappelons que *AC4* est un algorithme rapide qui fournit la consistance locale par rapport à des contraintes binaires du graphe étiqueté qui lui est donné en entrée.

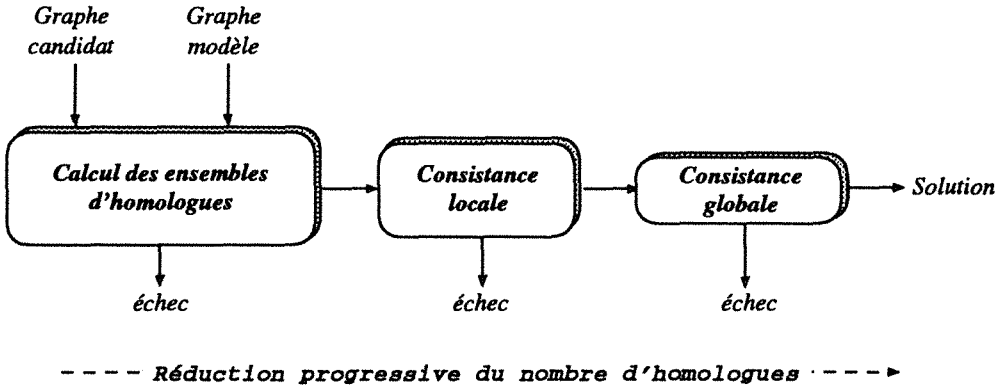


Figure 4.20. *Architecture générale de notre stratégie de mise en correspondance.*

La Figure 4.20 montre la structure générale de notre stratégie. Cette structure assure la réduction du nombre d'homologues par sommet modèle jusqu'à l'obtention d'un homologue par sommet (cas idéal). Le module du calcul des homologues ou *"étiquetage"* permet de déterminer pour chaque sommet du modèle la liste des homologues du candidat vérifiant la contrainte de sélection de correspondants potentiels (Contrainte 1). Le module de la recherche de la consistance locale élimine les homologues localement inconsistants, ceci réduit le nombre initial d'homologues obtenus par le premier module. Sur cet ensemble réduit d'homologues, une étape de recherche de la consistance globale est effectuée permettant d'éliminer les homologues globalement inconsistants. Ces deux modules de recherche de consistance sont détaillés dans la suite.

4.6.3 Consistance locale des homologues

Rappelons que notre problème de mise en correspondance a été formalisé comme étant un problème de propagation de contraintes à travers le graphe étiqueté $(\mathcal{V}, \mathcal{E}, \mathcal{H}_i)$ avec $|\mathcal{V}| = v$, $|\mathcal{E}| = e$ et $|\cup \mathcal{H}_i| = h$ (§ 4.4.4). Une application de l'algorithme de

$AC4$ réalise la propagation des contraintes et l'élagage des homologues inconsistants localement. Cet algorithme est présenté en détail dans l'annexe A. Afin de minimiser la complexité de notre approche, nous proposons de ne propager que les contraintes définies sur les couples de *sommets adjacents* du modèle pour la recherche de la consistance locale [Habacha 91a]. Présentons pour commencer les raisons de ce choix.

Rappelons que la complexité de $AC4$ est bornée par $O(ae^2)$ où a nombre d'arcs dans le graphe et e nombre d'étiquettes. Dans notre cas la complexité de $AC4$ est alors bornée par $\mathcal{O}(eh^2)$ or $e = a + a'$ où a est le nombre d'arête dans le modèle. Pour minimiser cette complexité, nous proposons de tester la consistance locale des homologues par rapport aux contraintes définies sur les couples de sommets adjacents dans le modèle. Ainsi, nous obtenons une complexité bornée par $O(ah^2)$. Le nombre d'homologues h' restants après l'application de $AC4$ est inférieur à h. Ce choix des couples de sommets *adjacents* est fait puisque il y a deux contraintes (Contrainte 2 et Contrainte 3'.(i)) définies sur les sommets adjacents contrairement au sommets non adjacents où il n'y a qu'une seule contrainte (Contrainte 3'.(ii)). Par conséquent, les contraintes propagées à travers $\mathcal{A}$ sont plus fortes et plus contraignantes puisque en plus de la disposition relative nous exigeons que la connexité soit conservée. Par conséquent, d'une part, nous minimisons la complexité de cette étape de consistance locale et d'autre part, la consistance globale est faite sur un ensemble réduit d'homologues ($h' < h$). Avant d'étudier le problème de la consistance globale des homologues, nous présentons la fonction de $Compatibilite\_adj(n_1, h_1, n_2, h_2)$ à propager par $AC4$ et qui vérifie la contrainte de connexité (Contrainte 2) et le test de cohérence de correspondance d'arêtes (Contrainte 3'(i)) pour deux couples d'homologues (n_1, h_1) et (n_2, h_2) dont les deux sommets n_1 et n_2 sont adjacents.

Soient $s = (t_{seg}; \{n_1, n_2\}; \rho)$ une arête du modèle, un homologue h_1 du sommet modèle n_1 tel que $h_1 = (n, [\ldots, a, c, \ldots])$ avec l'arête a correspond à l'arête s et un homologue h_2 du sommet modèle n_2 tel que $h_2 = (n', [\ldots, b, d, \ldots])$ avec l'arête b correspond à l'arête s. La fonction réalisant la vérification des contraintes 2 et 3'(i) est la suivante :

Fonction *Compatibilite_adj*(n_1, h_1, n_2, h_2)
  Cas a,b dans
    $a = b$ *et* $a \neq \Omega$:
      /* Les deux homologues sont localement consistante */
      retour(succès)
    $a = b$ *et* $a = \Omega$:
      Si $(Angle(\overrightarrow{nn'}, c) \approx Angle(a, c)$ *et* $Angle(\overrightarrow{n'n}, d) \approx Angle(b, d))$
      alors /* homologues localement consistants */
        retour(succès)

Sinon /* homologues non localement consistantes */
retour(échec)
$a \neq b$ *et* $a \neq \Omega$ *et* $b \neq \Omega$:
$e_1 = Ext\_seq\_droite(a)$
$e_2 = Ext\_seq\_droite(b)$
Si $e_1 = e_2$
alors/* homologues localement consistants */
retour(succès)
Sinon /* $e_1 \neq e_2$ */
Si $(Angle(\overrightarrow{e_1n}, \overrightarrow{e_1e_2}) \approx \pi$ *et* $Angle(\overrightarrow{e_2n'}, \overrightarrow{e_2e_1}) \approx \pi)$
alors/* homologues localement consistants */
retour(succès)
Sinon/* homologues non localement consistants */
retour(échec)
$a \neq b$ *et* $a = \Omega$:
$e_2 = Ext\_seq\_droite(b)$
Si $(Angle(\overrightarrow{e_2n'}, \overrightarrow{e_2n}) \approx \pi$ *et* $Angle(\overrightarrow{ne_2}, c) \approx Angle(a, c))$
alors/* homologues localement consistants */
retour(succès)
Sinon/* homologues non localement consistants */
retour(échec)
$a \neq b$ *et* $b = \Omega$:
/* cas est symmétrique au précédent */

Fin Fonction

La fonction $Ext\_seq\_droite(a)$ détermine l'extrémité de la chaine de segments approximant un segment de droite et commençant par a.

Remarque : Cette version de l'algorithme traite uniquement la consistance d'arc dans le cas où les deux sommets n_1 et n_2 du modèle sont adjacents par un segment de droite. Pour traiter le cas de l'arc de cercle, nous devons tout d'abord développer une fonction qui détermine la chaîne de segments reliant n et n' et qui vérifie si cette chaîne approxime un arc de cercle ayant le même rayon de courbure que celui du modèle.

La Figure 4.21 illustre les cas de consistance décrits par la fonction de compatibilité $Compatibilite\_adj$. Les deux sommets n_1 et n_2 sont appariés respectivement avec n et n'. Le segment s est mis en correspondance avec a dans h_1 homologue de n_1 et avec b dans h_2 homologue de n_2. Dans les différents cas de figures, les arêtes a et b appartiennent à la la même chaîne de segments approximant le segment s. Par contre la Figure 4.22 montre un exemple d'inconsistance. Nous constatons que les

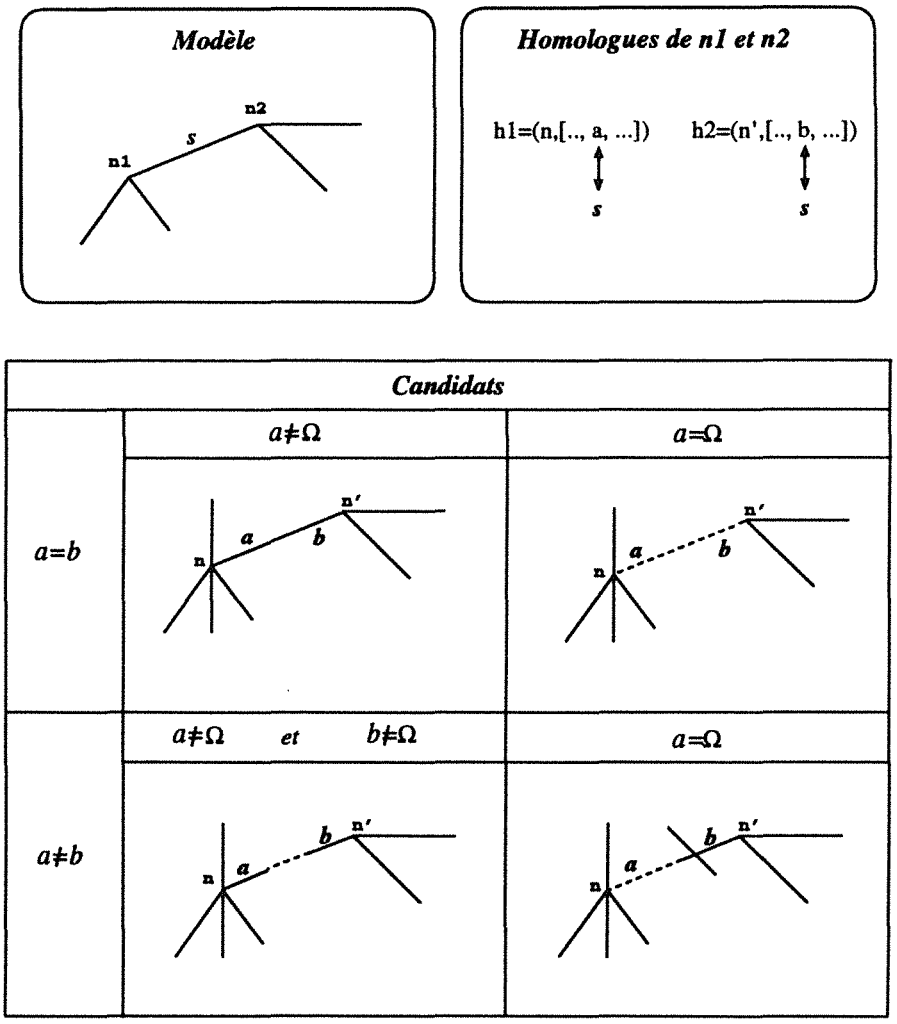


Figure 4.21. *Cas de consistance d'homologues.*

deux segments a et b ne peuvent pas appartenir à une même chaîne reliant n et n' et approximant un segment de droite. Par conséquent, n_1 apparié avec n en mettant en correspondance s avec a et n_2 apparié avec n' en mettant en correspondance s avec b sont incompatibles.

4.6.4 Consistance globale de la solution

Prouver la consistance globale d'un graphe revient à trouver un chemin consistant dans le graphe modèle de taille égale au nombre de sommets modèles. Pour cela, nous avons développé un algorithme de recherche de chemin consistant dans l'arbre de correspondance construit sur un ensemble réduit d'homologues [Habacha 92]. En effet, tous les homologues localement inconsistant ont été eliminés par *AC4*. Chaque

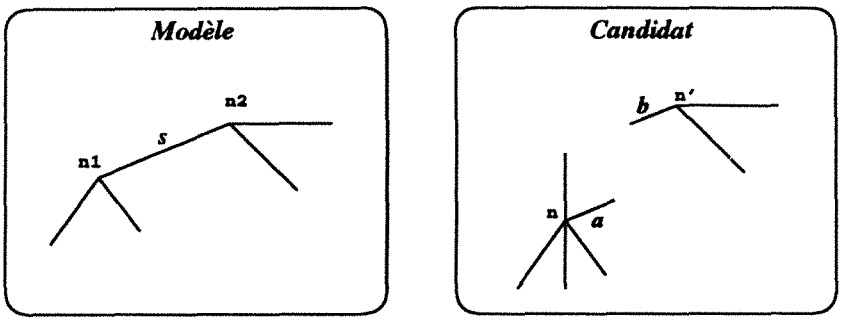


Figure 4.22. *Cas d'inconsistance d'homologues.*

nœud de cet arbre est un couple de la forme (sommet modèle, homologue). La consistance d'un chemin de l'arbre est définie par réccurrence.

Définition 7 *Soient C_i une branche de l'arbre consistante utilisant i sommets du modèle, n_{i+1} le $(i+1)^{ème}$ sommet du modèle et h_{i+1} un homologue de n_{i+1}.*

$$C_{i+1} = C_i \cup \{(n_{i+1}, h_{i+1})\}$$

$$si\ \forall k \in [1..i] \begin{cases} (n_k, h_k)\ et\ (n_{i+1}, h_{i+1}) vérifient \\ la\ Contrainte\ 2\ et\ la\ Contrainte\ 3' \end{cases} \tag{4.5}$$

Cette condition traduit le fait que le couple (n_{i+1}, h_{i+1}) est consistant avec tous les nœuds de la partie du chemin déjà construite. La solution du problème est le chemin C_N où N est le nombre de sommets du modèle.

Si la condition 4.5 n'est pas vérifiée pour h_{i+1}, alors le chemin C_{i+1} constitue une branche inconsistante et par conséquent, il faut passer à un autre homologue de n_{i+1}. Si nous avons essayé avec tous les homologues de n_{i+1} et dans tous les cas nous obtenons une inconsistance du chemin C_{i+1} alors un retour arrière est provoqué et par conséquent, il faut remettre en cause le chemin C_i. En d'autres termes, $\forall\ h$ *homologues de* n_{i+1}, $\exists k \in [1 \ldots i]$ (n_k, h_k) *et* (n_{i+1}, h) sont inconsistants alors dans C_i il faut enlever (n_i, h_i) et essayer de recalculer C_i en utilisant un autre homologue de n_i. Dans ce cas c'est un retour arrière à un seul niveau (du niveau $i+1$ (C_{i+1}), nous revenons sur le calcul du niveau i (C_i)).

Notons qu'un retour arrière de plusieurs niveau se produit si tous les homologues de n_{i+1} sont inconsistants avec le même nœud du chemin C_i déjà construit. A ce moment nous avons un retour arrière de $(i+1-l)$ où l est le niveau de nœud de C_i provoquant cet inconsistance. Ceci revient à dire que l'homologue du sommet modèle n_l utilisé dans la construction du C_i ne peut pas appartenir à la solution. La condition suivante exprime ceci :

$\exists\ l\ /(n_l, h_l) \in C_i,\ \forall\ h$ *homologue de* n_{i+1} (n_l, h_l) *et* (n_{i+1}, h) *sont inconsistants*

L'homologue h_l est alors supprimé de la liste d'homologues du sommet n_l. Notons que cet élagage est puissant ce qui nous permet de ne pas explorer tout l'arbre.

Du fait que nos contraintes dépendent de l'information d'adjacence entre les sommets du modèle, lors de l'implantation, nous exprimons la condition 4.5 par cas suivant que les sommets du modèle sont adjacents ou pas. Par conséquent, nous obtenons la condition suivante:

$$C_{i+1} = C_i \cup \{(n_{i+1}, h_{i+1})\}$$

$$si\ \forall k \in [1..i] \begin{cases} Compatibilite\_adj(n_k, h_k, n_{i+1}, h_{i+1}) = succès \\ \quad si\ n_k\ et\ n_{i+1}\ sont\ adjecents \\ Compatibilite\_non\_adj(n_k, h_k, n_{i+1}, h_{i+1}) = succès \\ \quad si\ n_k\ et\ n_{i+1}\ sont\ non\ adjecents \end{cases} \tag{4.6}$$

où $Compatibilite\_adj(n_1, h_1, n_2, h_2)$ teste si (n_1, h_1) et (n_2, h_2) vérifient la Contrainte 2 et la Contrainte 3'(i) (cf 4.6.3) et $Compatibilite\_non\_adj(n_1, h_1, n_2, h_2)$ teste si (n_1, h_1) et (n_2, h_2) vérifient la Contrainte 3'(ii).

Remarquons qu'au moment de l'élagage des homologues localements inconsistants par AC4, nous mettons à jour une structure contenant toutes les correspondances consistantes par rapport à la Contrainte 2 et la Contrainte 3'(i). Par conséquent, pour les couples de sommets adjacents, la vérification de la condition 4.6 revient juste à vérifier l'appartenace du couple à cette structure. Soulignons également que l'arbre n'est jamais complètement crée, seule la branche courante du chemin est effectivement construite.

4.6.5 Vérification du résultat de la mise en correspondance

Une étape de vérification du résultat de la mise en correspondance est nécessaire; elle consiste à vérifier si les points reconnus comme points de connexion appartiennent réellement à des lignes de connexion dans le schéma complet. Cette vérification utilise la connaissance de ligne de connexion qui a permis auparavant la sélection du graphe candidat.

La Figure 4.23 montre un exemple de mauvaise reconnaissance. En effet, la partie en gras du candidat est mis en correspondance avec la capacité en considérant les deux points n et n' comme étant des points de connexion. Or le symbole candidat a été selectionné en utilisant la ligne de connexion L déjà détectée au paravant et par conséquent, le symbole reconnu doit être relié au schéma à travers cette ligne de

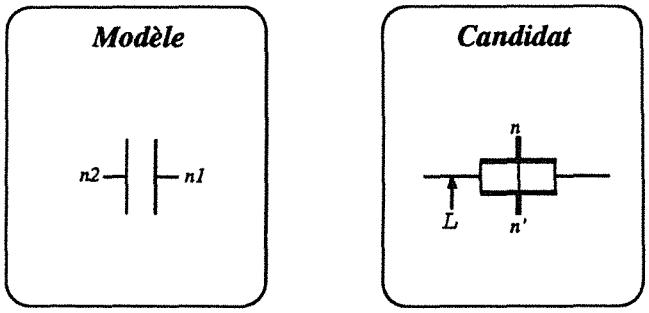


Figure 4.23. *Nécessité de la vérification du résultat de la mise en correspondance.*

connexion. Ce qui n'est le cas puisque cette mise en correspondance suppose que la ligne L est du bruit.. Ceci provoque un rejet du résultat de la mise en correspondance et par conséquent, le modèle n'est pas retenue pour ce candidat.

4.6.6 Résultats expérimentaux

Nous venons de détailler la mise en correspondance d'un symbole modèle avec un symbole candidat. Dans cette section, nous validons notre approche par différents résultats expérimentaux. L'organisation de cette section permet de montrer la prise en compte du bruit.

A- Correction du symbole candidat en ajoutant les segments manquants

La Figure 4.24 montre un exemple de résultat expérimental. l'image binaire correspondant à la zone supposée sélectionnée par le processus de construction d'hypothèses de présence de symboles est présentée dans la Figure 4.24.a. La Figure 4.24.b montre l'ensemble des segments obtenus après application des processus du bas niveau. Nous pouvons facilement constater que le symbole est affecté par le bruit. En effet, il existe des segements n'appartenant pas au symbole recherché. Nous remarquons également qu'il lui manque un segment. Après la mise en correspondance du symbole candidat avec le modèle, nous obtenons le résultat donné par la Figure 4.24.e. Remarquons que tous les segments n'appartenant pas à la partie du candidat appariée avec le modèle sont supposés être en trop. Ces segments proviennent du fait que la zone candidate ne contient pas exactement le symbole recherché. Signalons également que le segment $(27, 28)$ du modèle (Figure 4.24.c) est apparié avec le segment fictif Ω puisque son correspondant est manquant. Une correction du symbole est alors effectuée en ajoutant ce segment au symbole candidat. Nous notons que dans cet exemple, le candidat et le modèle ne possèdent ni la même taille, ni la même orientation (l'angle de rotation est égale à π).

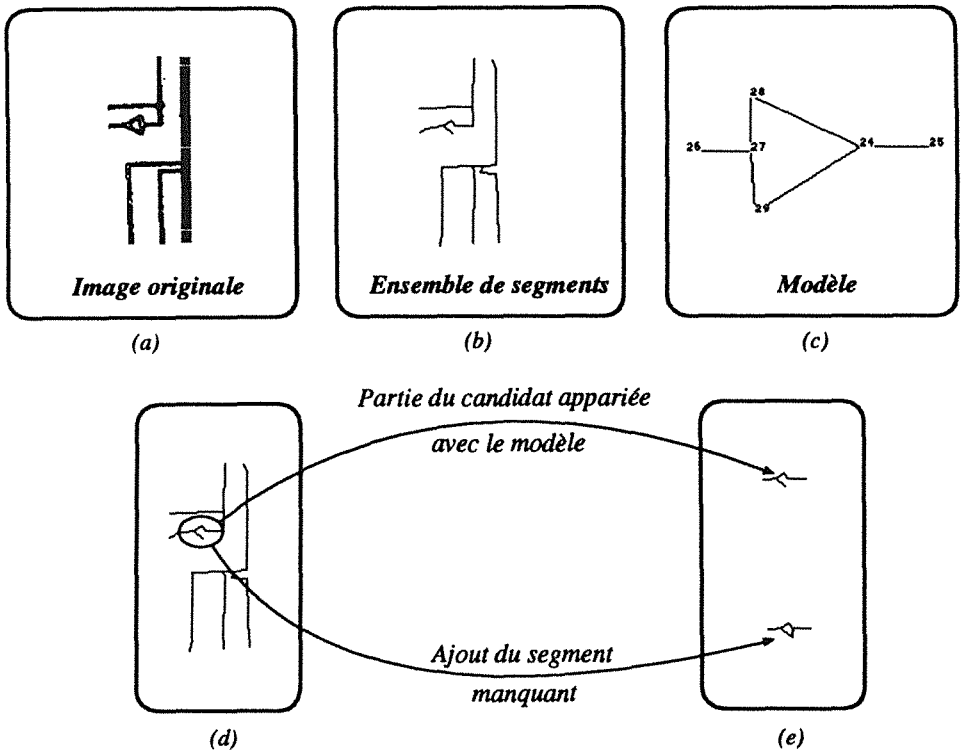


Figure 4.24. *Correction de la partie appariée du symbole candidat.*

B- Elimination des segments érronés

Les Figures 4.25 et 4.26 montrent deux applications du processus de mise en correspondance où il y a eu élimination de segments produits par l'étape de squelettisation de l'image. L'image originale se trouve dans les figures *(a)*, la deuxième présente le résultat du processus du bas niveau. les Figure 4.25.e et 4.26.e montre la partie du symbole candidat mis en correspondance avec le modèle. Nous constatons que après l'étape de squelettisation, il y a eu apparition de barbules qui ont été éliminés lors de l'appariement. Remarquons qu'à part ce type d'erreurs, il existe plusieurs segments qui n'ont pas été appariés puisqu'ils n'appartiennent pas au symbole recherché. Par conséquent, bien que la zone sélectionnée n'est pas parfaite, la mise en correspondance n'est pas affectée. Remarquons également que le modèle et le candidat dans ces deux cas de figures possèdent la même taille mais n'ont pas la même orientation. En effet, il existe un angle de $\frac{\pi}{2}$ entre les deux symboles.

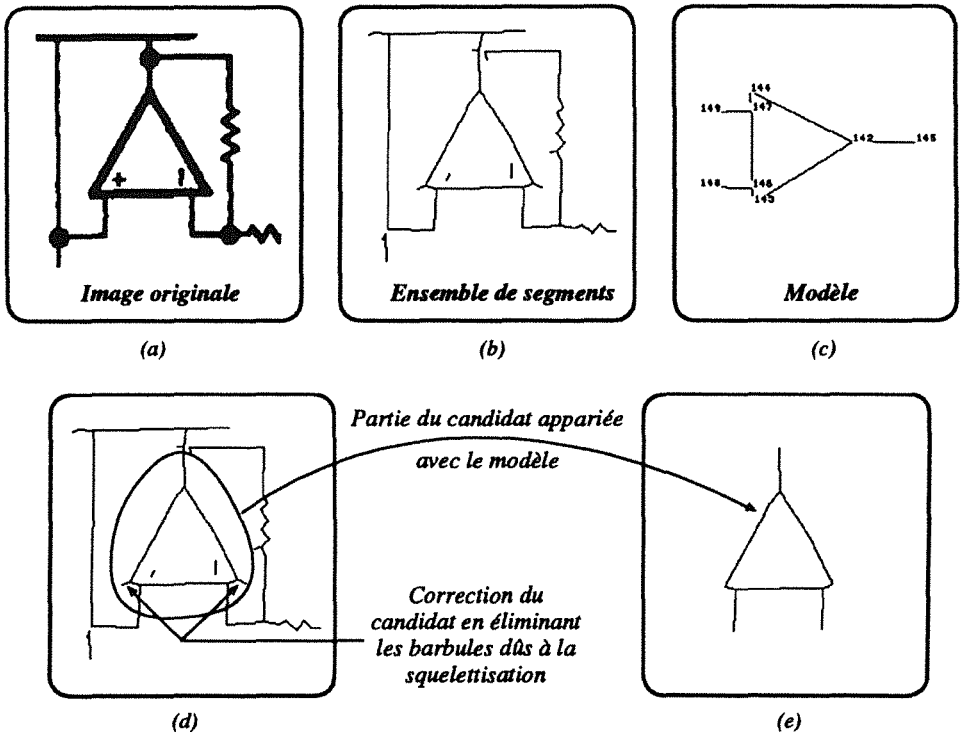


Figure 4.25. *Elimination des barbules lors de l'appariement d'un amplificateur.*

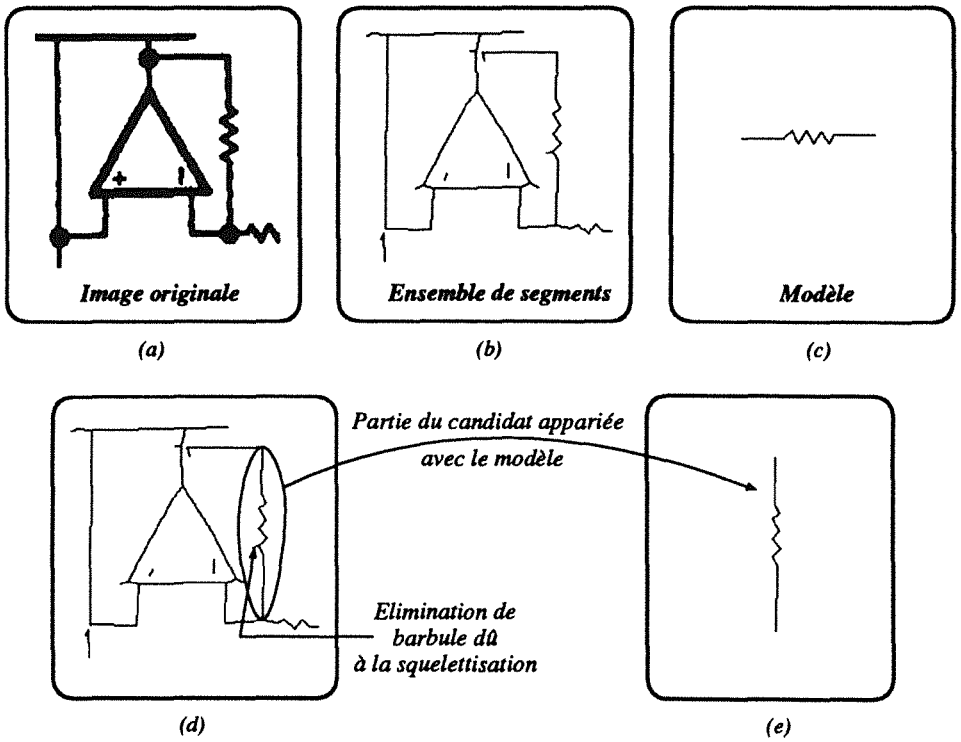


Figure 4.26. *Elimination des barbules lors de l'appariement d'une résistance.*

C- Cas d'échec de la mise en correspondance

La Figure 4.27 montre un cas d'échec du processus de mise en correspondance. Après application des processus du bas niveau, les deux extrémités de la résistance ont été déformées. En effet, la partie entourée de la Figure 4.27.d correspond à celle entourée du modèle (Figure 4.27.e). Mais, étant donné que les angles entre les segments sont complètement perdus, cette mise en correspondance ne peut être effectuée. Ceci provient du fait que les erreurs sur les petits segments se répercutent sur leurs directions en s'amplifiant. Ce qui explique que les angles soient complètement erronés.

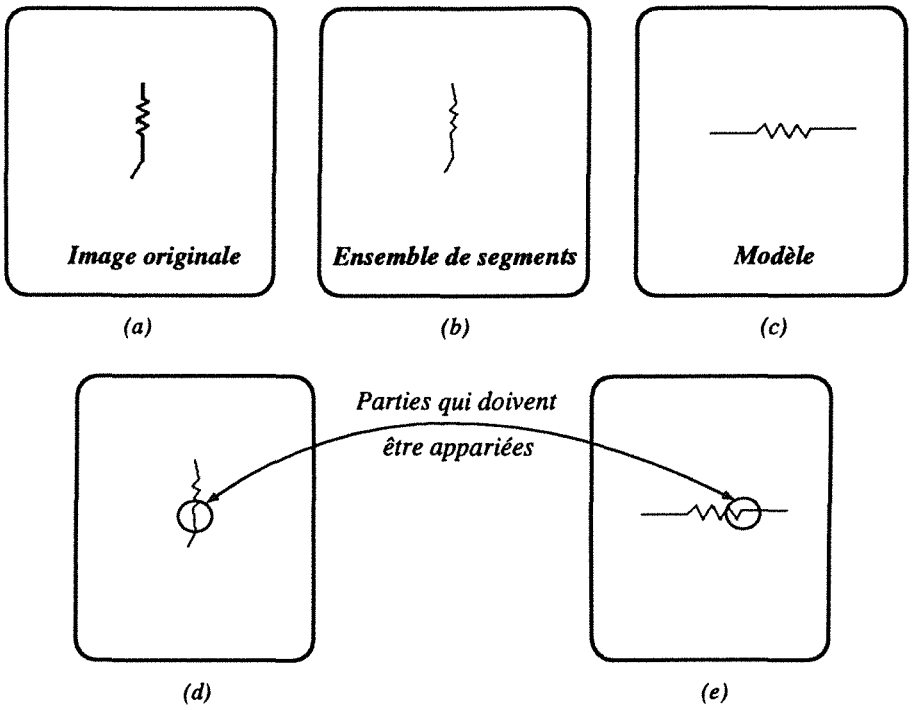


Figure 4.27. *Cas d'échec de la mise en correspondance.*

Remarquons que les Figures 4.26 et 4.27 n'ont pas été saisies avec la même résolution. Pour la première la résolution est de 300 points par pouce, par contre la deuxième est de 150 points par pouce. Ce qui explique en partie cette déterioration.

4.7 Choix du meilleur modèle

Dans le cas où un graphe candidat est apparié avec plusieurs modèles, une étape de choix du modèle est nécessaire. Cette étape consiste à choisir le modèle réalisant le *"meilleur appariement"* que nous caractérisons par le maximum d'appariememt exact. Nous distinguons trois cas :

- le cas où les parties du symbole candidat, appariées avec des modèles différents, sont disjoints. Ceci indique que la zone candidate sélectionnée contient plus d'un symbole et dans ce cas tous les modèles sont retenus. Les Figures 4.25 et 4.26 montrent un exemple de symbole candidat où les deux parties appariées respectivement avec l'amplificateur et la résistance sont disjointes. Les deux modèles sont alors retenus.

- le cas où les parties du symbole candidat ne sont pas disjointes, nous choisissons le modèle de taille supérieur en nombre de sommets. Ceci se produit dès qu'un modèle est inclue dans un autre modèle. En effet, dans la Figure 4.28, le candidat est mis en correspondance avec la diode en supposant que le segment s est du bruit. Ce candidat est aussi apparié avec le thyristor. Dans ce cas, le modèle retenu est le thyristor.

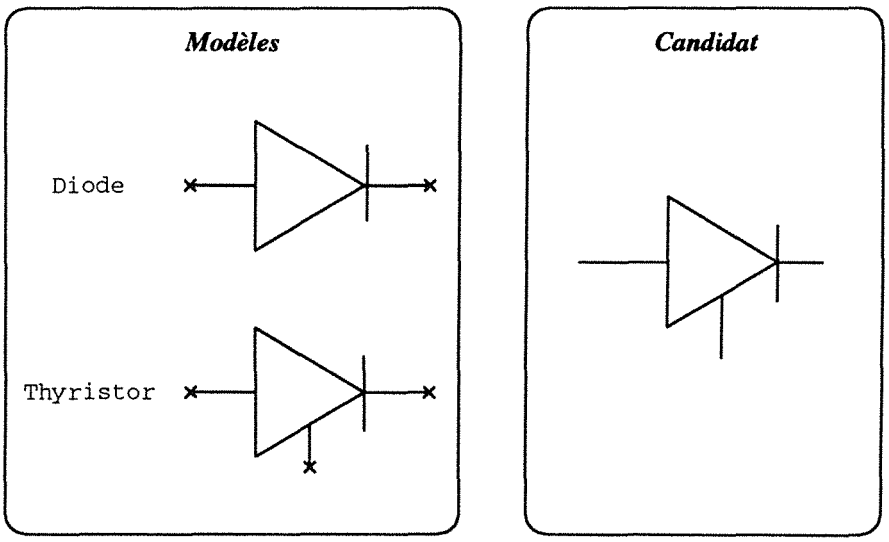


Figure 4.28. *Choix du meilleur modèle en comparant les tailles des modèles appariés.*

- le cas où les parties du candidat ne sont pas disjointes et les modèles reconnus sont de même taille, nous choisissons le modèle ayant le maximum de sommets réalisant un appariememt exact et ayant le minimum d'arêtes supposées manquantes au candidat.

Remarque Notons que le nombre de modèle restant est assez réduit, puisque dès qu'il y a une inconsistance, le modèle est rejeté dans l'une des étapes précédant le choix du meilleur modèle.

4.8 Discussion

Nous avons présenté dans ce chapitre un processus de reconnaissance d'un symbole candidat fondé sur la *mise en correspondance structurelle* de graphes attribués. Rappelons qu'un symbole candidat est la zone du schéma sélectionnée par l'étape de construction d'hypothèse de présence de symboles.

Pour élaborer ce processus de reconnaissance, nous avons proposé trois contraintes permettant de construire les appariements et d'éliminer les incohérences des homologues obtenus :

- une contrainte de *sélection de correspondants potentiels* qui détermine l'ensemble des sommets candidats homologues pour chaque sommet du modèle en utilisant l'information topologique et géométrique,
- une contrainte de *connexité* qui vérifie la conservation de la structure topologique du modèle dans le candidat,
- une contrainte de *disposition relative* qui vérifie la préservation de la structure géométrique du modèle dans le candidat.

Nous avons montré que ces trois contraintes sont invariantes par rapport à la rotation et l'homothétie. Ce qui nous permet de définir un processus de reconnaissance indépendant de l'orientation et de la taille du modèle par rapport à celle du candidat. Par ailleurs, ces contraintes permettent de tenir compte du bruits inhérents aux processus de bas niveau d'une part et ceux introduit par l'étape de construction d'hypothèses de présence de symboles d'autre part. Ainsi, le bruit affectant les sommets est traité par la contrainte de sélection de correspondants potentiels, par contre celui affectant les arêtes est pris en compte par la contrainte de connexité. La contrainte de sélection de correspondants potentiels permet de traiter deux sortes de bruit : d'une part les arêtes manquantes au sommet et d'autre part, les arêtes du candidat qui sont en trop. Cela dit, il faut relever le fait qu'en aucun cas nous n'autorisons la combinaison de ces deux types de bruit. Cette restriction a été établie par un choix, puisque si cela était permis, tout sommet du modèle pourrait être apparié avec tous les sommets du candidat. En effet, il suffit d'ajouter les arêtes fictives Ω qui seront appariées avec les arêtes du sommet modèle n'ayant pas de correspondants et d'enlever toutes les autres arêtes du sommet candidat en supposant qu'elles sont en trop. C'est pour cette raison que nous avons décidé de ne pas admettre un taux d'erreurs trop élevé et de n'accepter qu'un type de bruit à la fois.

La représentation des symboles par des graphes attribués est facilement enrichissable par des attributs numériques ou symboliques. En effet, par exemple si un segment admet comme attribut symbolique la position de la zone noire (gauche

ou droite) qu'il borde, il suffit de le représenter par une arête ayant un nouveau définissant la position de la zone noire par rapport au segment. Au niveau de l'appariement, ces nouveaux attributs seront traités dans la contrainte de connexité qui préserve les information topologiques. Comme, nous exigeons que l'adjacence soit faite avec une arête qui a les mêmes attributs que l'arête du modèle, il suffit alors de ne plus se limiter à tester l'adjacence avec uniquement le même type mais de l'étendre de sorte à tenir compte de toute l'information disponible. Remarquons que dans la version actuelle de notre système, la prise en compte de ces informations est prévue et pourra donc être utilisée à la donnée de ces informations par le bas niveau. L'évolution du processus de reconnaissance pour traiter les symboles contenant des zones noires est alors faisable et constitue une de nos perspective.

Avant de finir cette discussion, il ne faut pas oublier de mentionner le problème d'indexation de la base des modèles ; dans l'approche réalisée, une mise en correspondance avec tous les modèles de la base est effectuée, puis le meilleur modèle est retenu. Cependant, comme la méthode développée est hiérarchique, elle permet de réduire la complexité de l'étape de mise en correspondance. Par ailleurs, le processus de reconnaissance développé est parallélisable puisqu'il est possible de réaliser l'étape de mise en correspondance en parallèle pour tous les modèles. Ceci ne réduit pas pour autant l'intérêt que nous portons à ce problème. Mais l'objectif que nous nous sommes fixé est de réaliser tous les modules permettant d'analyser un schéma complet. Pour cette raison, nous avons décidé de traiter en priorité le problème de détection des lignes de connexion, ainsi que celui de construction d'hypothèses de présence de symboles que nous avons supposé fait dans le présent chapitre. Le problème d'indexation de la base de modèles est alors abordé comme étant une des perspectives de ce travail (chapitre 6).

5
Analyse contextuelle de schémas complets

Rappelons que le processus d'interprétation de schémas réalise deux tâches : la détection des lignes de connexion d'une part et celle des symboles d'autre part. Cette dernière tâche est fondé sur le principe de *prédiction-vérification d'hypothèses.* La prédiction consiste à la sélection d'une zone susceptible de contenir un symbole et la vérification consiste à la reconnaissance du symbole contenu dans cette zone.

Dans ce qui précède, nous avions étudié et résolu le problème de reconnaissance en supposant qu'une étape de construction d'hypothèses de présence de symboles est réalisée (chapitre 4). A présent, il s'agit, d'expliquer cette étape de construction d'hypothèses, de détailler le processus de détection des lignes de connexion pour enfin décrire le fonctionnement général de notre système d'analyse de schémas.

Ce chapitre présente, dans un premier temps, l'architecture détaillée du processus d'interprétation (section 5.1). Ensuite, une description du processus de construction d'hypothèses de présence de symboles est effectuée dans section 5.3. Nous détaillons alors l'étape de détection des lignes de connexion (section 5.4). Puis, nous présentons dans la section 5.5 le fonctionnement du système, qui consiste à gérer les échanges des différents modules constituant le processus d'interprétation. Enfin la section 5.6 est consacrée à la présentation des résultats expérimentaux.

5.1 Méthodologie du module d'interprétation

L'idée clé adoptée pour l'interprétation consiste à tirer profit des *liens contextuels* qui existent entre les informations texte, lignes de connexion et symboles d'un même schéma. Ces liens se traduisent par la forte probabilité de présence de symboles au voisinage du texte et aux extrémités des lignes de connexion et réciproquement les lignes reliant un symbole au schéma sont des lignes de connexion. A notre connaissance, ces liens contextuels n'ont jamais été exploités dans les travaux d'analyse de

schémas.

La Figure 5.1 décrit l'architecture du processus d'interprétation de notre système. Les primitives segments, points de jonctions, composantes textes et boucles caractéristiques sont extraites par les processus du bas niveau déjà décrits dans § 3.2. Ce processus d'interprétation est constitué de trois modules : détection des lignes de connexion, construction d'hypothèses de présence de symbole et reconnaissance de symboles. Les deux derniers modules constituent la phase de détection de symboles.

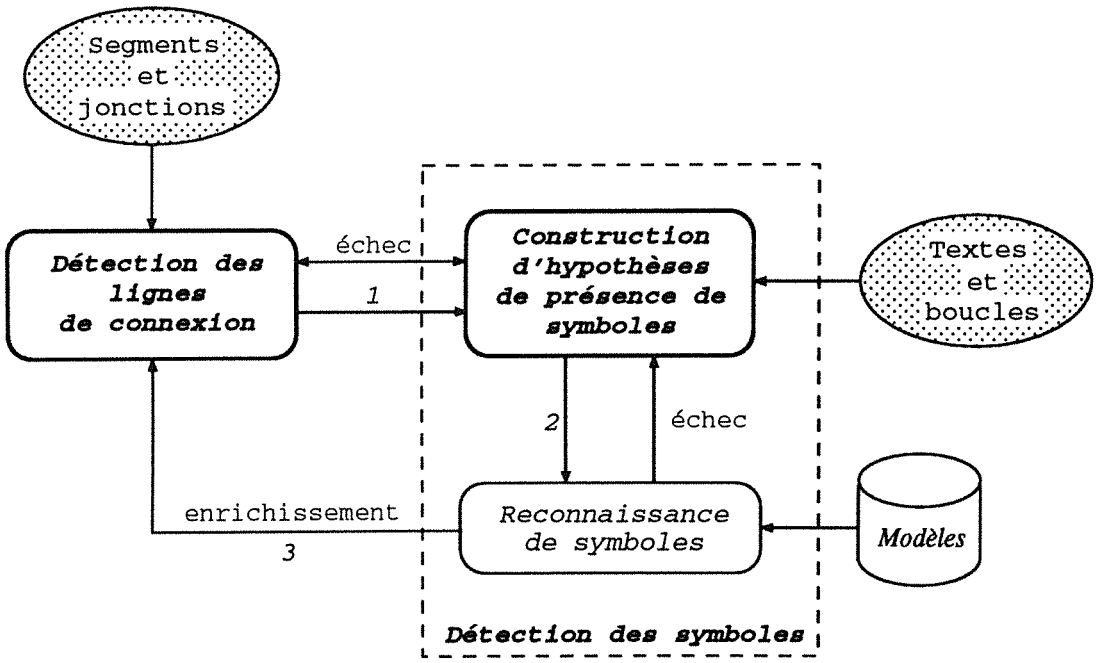


Figure 5.1. *Architecture générale du processus d'interprétation.*

Nous constatons que ces trois modules forment une boucle *(1 2 3)* qui traduit le lien contextuel qui existe entre les informations lignes de connexion et symboles. En effet, l'idée générale que nous avons adoptée pour le processus d'interprétation consiste à sélectionner à partir des lignes de connexion et à partir des critères que nous avons établi une zone du schéma constituant l'hypothèse de présence de symbole dans cette zone.

L'étape de construction d'hypothèses est fondée en priorité sur la forte possibilité de présence de symboles aux extrémités des lignes de connexion. Cette hypothèse de présence de symbole à une extrémité d'une ligne de connexion est appuyée par la présence du texte ou de boucles caractéristiques au voisinage de ce point. Dans le cas où tous les points extrémités ont été utilisés et aucun nouveau point n'est détecté, l'étape de construction d'hypothèses utilise alors les boucles caractéristiques non traitées auparavant. Ce deuxième traitement permet de remédier au problème du bruit qui peut engendrer l'existence de symboles ou de parties de schéma déconnectés

du schéma complet et qui sont par conséquent, non atteint par un simple suivi des lignes de connexion.

Remarquons que l'utilisation en priorité de l'information points extrémités des lignes de connexion par rapport à l'information boucles caractéristiques, permet de réduire le nombre de fausses boucles traitées. Rappelons qu'une fausse boucle est une boucle engendrée par l'intersection des lignes de connexion et par conséquent, une partie de son contour est une ligne de connexion.

Une fois la zone de présence construite, deux cas peuvent se présenter :

- soit l'hypothèse de présence de symbole est confirmée par le module de reconnaissance et par conséquent, de nouvelles lignes de connexion sont détectées et ce à partir des points du symbole reconnus comme étant des points de connexion. Ainsi nous procédons donc à un *enrichissement* de l'ensemble des lignes de connexion. Remarquons que cet enrichissement est *totalement guidé* par la reconnaissance de symboles.

- soit la reconnaissance échoue et dans ce cas une remise en cause du résultat de construction d'hypothèses est activée. Ceci se traduit généralement par un *agrandissement de la zone* ce qui permet de pallier au problème de sélection d'une zone contenant une sous partie d'un symbole. En effet, le module de construction des hypothèses de présence de symboles peut ne pas donner, ne serait ce qu'à cause du bruit, un symbole candidat parfait. En fait, en cas de litige trois cas de figures peuvent se présenter dont deux sont pertinents soit la zone sélectionnée contient des segments en plus de ceux appartenant au symbole, soit nous obtenons une zone contenant une partie du symbole ou bien la zone fournie ne correspond à rien. Si le premier cas est traité par la phase de reconnaissance il reste à résoudre le deuxième cas et c'est pour ceci que nous proposons d'agrandir la zone.

Soulignons que dans le cas où aucune zone candidate n'a pu être sélectionnée, ce qui se traduit par un échec de la construction d'hypothèses, une détection de nouvelles lignes de connexion par suivi à partir de points particuliers sans utiliser la reconnaissance de symboles est effectuée (§ 5.4).

Il importe néanmoins de remarquer que la forte coopération entre les trois modules, détection des lignes de connexion, construction des hypothèses et reconnaissance de symboles, implique que nous devons prévoir un traitement d'initialisation permettant *l'activation du système.* Cette activation peut être réalisée de deux manières :

1. soit en activant le processus de la détection des lignes de connexion, ce qui se traduit par l'initialisation de l'ensemble des lignes de connexion.

2. soit en activant le processus de construction des hypothèses de présence de symboles, qui se traduit par la sélection d'une zone candidate sans utiliser l'information lignes de connexion.

Afin d'avantager l'utilisation des extrémités des lignes de connexion par rapport à celle des boucles dès l'activation du système, nous proposons d'activer notre système par une initialisation de l'ensemble des lignes de connexion si celle-ci est possible pour le schéma traité. Dans le cas où cette initialisation ne donne aucun résultat, l'activation du système est effectuée par le processus de construction d'hypothèses de présence de symboles. Remarquons que dans ce dernier cas, l'ensemble des lignes de connexion est indirectement initialisé par le biais de la reconnaissance de symboles.

Comme le processus de reconnaissance a été étudié dans le chapitre précédent, dans le présent chapitre, notre intérêt se porte sur la construction d'hypothèses de présence de symboles, la détection des lignes de connexion et le fonctionnement de la totalité du système, c'est-à-dire la gestion des différents processus constituant le haut niveau et les échanges entre eux. Avant de détailler ces différents modules, nous commençons par définir l'ensemble de points particuliers appartenant aux lignes de connexion.

5.2 Points particuliers d'une ligne de connexion

Après vectorisation du schéma les lignes de connexions se retrouvent généralement constituées d'ensembles de segments colinéaires. La Figure 5.2 montre un exemple qui illustre ce phénomène. En effet, la ligne de connexion L_1 est constituée de 4 segments et elle relie les deux points e_1 et e_2. Pour nous, une ligne de connexion est déterminée par le suivi d'un ensemble de segments colinéaires admettant deux extrémités.

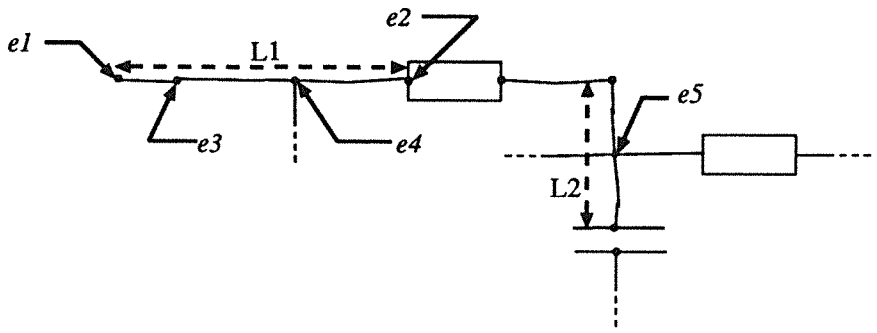


Figure 5.2. *Types de points appartenant à une ligne de connexion.*

Nous relevons lors de la formation des lignes de connexion trois types de points qui peuvent leurs appartenir :

- points de jonctions qui sont des extrémités des lignes de connexion, par exemple les deux points e_1 et e_2. Nous désignons dans la suite ces points par *points extrémités.*

- points de jonctions qui appartiennent aux lignes de connexion et qui sont différents des points extrémités. Ces points de degré strictement supérier à deux matérialisent les intersections entre lignes de connexion. Ils sont désignés par *points de branchement.* Par exemple le point e_4 est un point de branchement de la ligne de connexion L_1 et le point e_5 est un point de branchement de la ligne de connexion L_2.

- points de degré deux et qui relient deux segments colinéaires formant la ligne de connexion. Ces points générés par le processus du bas niveau sont sans intérêt pour l'analyse du schéma, par exemple le point e_3 de la Figure 5.2.

Dans la suite, nous verrons que les points extrémités interviennent dans le processus de construction d'hypothèses de présence de symboles. Par ailleurs, ils interviennent avec les points de branchement dans le processus de détection des lignes de connexion.

5.3 Construction d'hypothèses

La reconnaissance s'effectue après avoir sélectionné une zone candidate. Le processus de construction d'hypothèse de présence de symbole consiste donc à sélectionner une zone du schéma susceptible de contenir un symbole ; hypothèse qui sera confirmée ou remise en cause par la reconnaissance.

Rappelons que dans les différents travaux réalisés dans ce domaine [Groen 84, Lin 85, Okazaki 88], l'utilisation des boucles est faite à l'aveugle et de façon exhaustive. En effet, dans ces travaux, un parcours de toutes les boucles détectées, fausses ou pas, est réalisé et pour chaque boucle une étape de reconnaissance est accomplie. Pour éviter cette utilisation des boucles à l'aveugle et par conséquent, minimiser le nombre de fausses boucles traitées, nous proposons d'utiliser en priorité l'information *points extrémités* des lignes de connexion. En effet, pour délimiter une zone candidate, le module de construction d'hypothèse utilise le fait qu'un symbole ne peut appartenir au schéma complet que s'il est rattaché à une ligne de connexion. Ainsi, aux extrémités des lignes de connexion, il y a une forte probabilité de présence de symbole. C'est pour cette raison que nous tenons à jour l'ensemble des points extrémités des lignes de connexion détectées. Pour réaliser cette étape de prédiction de présence de symboles, nous définissons quatre règles de sélection de zones candidates.

5.3.1 Règles de sélection de zones candidates

S'il est vrai que la probabilité de l'existence d'un symbole à une extrémité de ligne de connexion est forte, il n'en reste pas moins qu'il faut déterminer la taille de la zone à étudier autour de cette extrémité et à renforcer cette hypothèse de présence de symboles. Nous proposons alors d'utiliser d'une part l'information boucle caractéristique et d'autre part l'information texte proches du point extrémité à étudier. L'information boucle permet de détecter la présence d'un symbole contenant des boucles et l'information texte permet de détecter les symboles qui ne comportent pas de boucles comme le cas d'une capacité. Rappelons que cette primitive *texte* n'est pas utilisée dans les travaux d'analyse de schémas.

Pour délimiter une zone succeptible de contenir un symbole au voisinage d'un point extrémité d'une ligne de connexion, l'idée consiste donc à utiliser l'existence des boucles ou celle du texte. Ainsi étant donné un ensemble E de points extrémités, nous définissons les deux règles suivantes permettant la sélection d'une zone candidate en utilisant, en plus des points extrémités des lignes de connexion, les boucles caractéristiques ou les composantes textes :

Règle (S_1) *il existe un point extrémité e dans E tel que e est connexe à une boucle caractéristique. Dans ce cas, la zone candidate est définie comme étant le rectangle qui inclut strictement le rectangle englobant de cette boucle caractéristique et le point extrémité e.*

La Figure 5.3 montre un exemple qui illustre la mise en application de la règle de sélection (S_1). Le rectangle en pointillé est le rectangle englobant de la boucle caractéristique B. Le point extrémité e de la ligne de connexion L est connexe à la boucle caractéristique B et par conséquent, la règle de sélection (S_1) est applicable et fournit la zone candidate donnée par le rectangle en gras.

Règle (S_2) *il existe un point extrémité e dans E tel qu'au moins une composante texte appartient à son voisinage. Dans ce cas la zone candidate est définie comme étant un rectangle ayant la taille moyenne des boucles caractéristiques du schéma, qui inclut strictement le point extrémité e et se situe dans le sens de parcours de la ligne de connexion de e.*

Remarque Dans un schéma donnée, les symboles possèdent des tailles comparables. En d'autres termes si la base des modèles est normalisée, tous les symboles du schéma seront définis par rapport aux modèles correspondants avec le même rapport d'homothétie. Pour cette raison que dans la règle (S_2), nous proposons de calculer la taille des zones candidates en fonction de la moyenne des tailles de toutes les boucles caractéristiques du schémas.

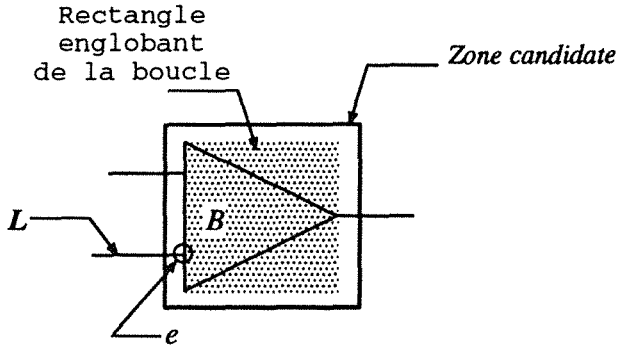


Figure 5.3. *Sélection de zones candidates utilisant les points extrémités et les boucles caractéristiques.*

La Figure 5.4.a montre un exemple qui illustre la mise en application de la règle de sélection (S_2). Il existe du texte au voisinage du point extrémité e de la ligne de connexion L et par conséquent, la règle (S_2) est applicable et fournit la zone candidate donnée par le rectangle en gras. La Figure 5.4.b montre un exemple de rectangle qui ne vérifie pas la condition de positionnement du rectangle par rapport à l'orientation de parcours de la ligne de connexion L donnée par la règle (S_2) et par conséquent, ce rectangle ne peut pas être la zone candidate.

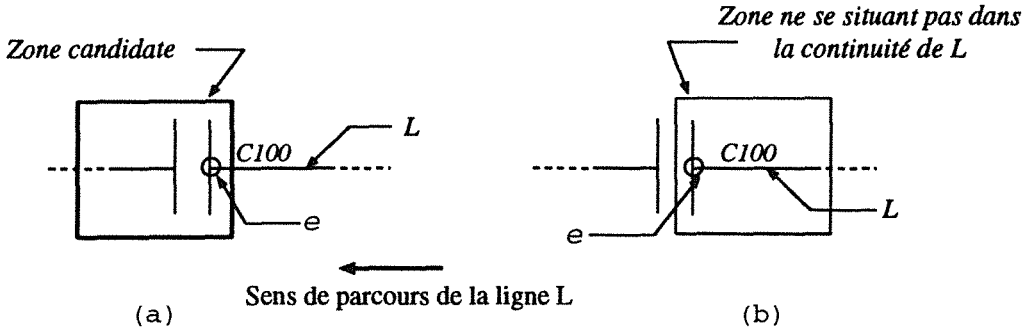


Figure 5.4. *Sélection de zones candidates utilisant les points extrémités et le texte.*

Dans les schémas l'information texte n'est pas toujours présente, et par conséquent, si l'existence du texte permet d'appuier la présence de symbole au voisinage du point extrémité, son absence ne doit en aucun cas mettre en cause la prédiction de la présence d'un symbole. Pour pouvoir détecter les symboles ne contenant pas de boucles et n'ayant pas du texte au voisinage, nous proposons une troisième règle de sélection de zone candidate qui utilise uniquement l'information point extrémité en étudiant son voisinage. Bien évidemment, cette règle est moins prioritaire que les deux précédentes puisque l'hypothèse de présence de symbole n'a été appuiyée par

aucune autre information et qui la rendrait plus plausible.

Règle (S_3) *il existe un point extrémité* e_1 *dans* E *tel que il existe un rectangle incluant le point extrémité* e_1*, ayant comme taille au moins la moyenne des tailles de toutes les boucles caractéristiques du schéma*[1] *et qui contient un ensemble de segments ainsi qu'un deuxième point extrémité* e_2 *d'une ligne de connexion déjà détectée.*

La condition d'existence d'un deuxième point extrémité permet de renforcer l'hypothèse de présence de symbole au premier point extrémité.

La Figure 5.5 montre un exemple utilisant uniquement l'information point extrémité. Au point extrémité e_3 la règle (S_3) n'est pas applicable puisque le rectangle ne contient aucun groupe de segments. Par contre au point extrémité e_1 cette règle est applicable puisque il existe un rectangle contenant un ensemble de segments et un deuxième point extrémité.

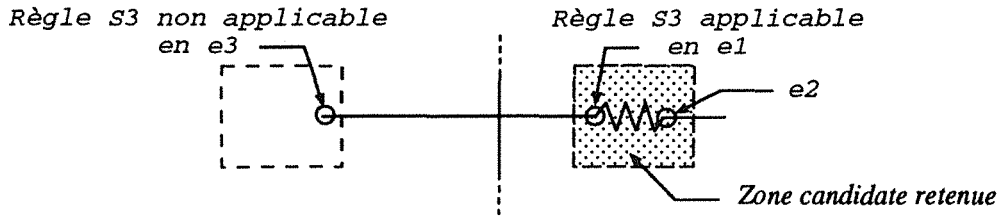


Figure 5.5. *Sélection de zones candidates utilisant uniquent les points extrémités.*

Remarque La règle (S_3) est dite *temporairement non applicable*, si la zone contient un ensemble de segments mais pas le deuxième point extrémité e_2. En effet, elle peut être déclenchée à un autre instant si au cours de l'interprétation, le deuxième point extrémité a été détecté. Dans le cas où cette règle est temporairement non applicable le point extrémité e_1 est gardé comme un point problématique et à l'activation de la règle (S_3), le deuxième point extrémité e_2 est recherché parmi tous les points extrémités.

Notons qu'utiliser en priorité les extrémités des lignes de connexion n'est pas suffisant pour délimiter tous les symboles existants dans le schéma. En effet, il faut tenir compte que dû au bruit certains symboles ou certaines parties du schéma peuvent se trouver isolés et par conséquent, ne peuvent être atteints en utilisant uniquement l'information points extrémités des lignes de connexion déjà détectées. C'est pour ceci, que nous avons enrichi l'ensemble des trois règles de sélection définies auparavant par une quatrième règle qui nous permet de pallier à cette faiblesse :

1. Le rectangle se situe dans le sens de parcours de la ligne de connexion de e_1.

Règle (S_4) *dans le cas où il existe une boucle caractéristique non utilisée par la règle* (S_1) *et non détectée comme une fausse boucle, la zone candidate est définie comme étant le rectangle qui inclut strictement cette boucle caractéristique.*

La Figure 5.6 montre un exemple qui illustre l'application de cette règle de sélection. Dans ce cas la résistance ne peut être atteint en suivant les lignes de connexion. La zone candidate est alors sélectionnée en utilisant la boucle caractéristique B.

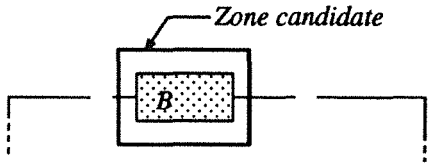


Figure 5.6. *Sélection d'une zone candidate utilisant uniquement les boucles caractéristiques.*

Comme nous l'avons déjà dit, un ordre de priorité d'application de ces règles a été instauré :

$$(S_1) > (S_2) > (S_3) > (S_4)$$

où $(S_i) > (S_j)$ exprime le fait que (S_i) est plus prioritaire que (S_j).

Ainsi, les règles (S_1), (S_2) et (S_3) sont plus prioritaires que la règle (S_4), ce qui permet d'avantager l'utilisation de l'information point extrémité par rapport à celle des boucles et par conséquent, minimiser le nombre de fausses boucles visitées. Remarquons que la priorité des règles (S_1), (S_2) et (S_3) par rapport à la règle (S_4) se traduit par l'activation de cette dernière règle si et seulement si tous les points extrémités ont été traités et par conséquent, l'ensemble E des points extrémités est vide.

La Figure 5.7 résume la structure du processus de construction d'hypothèses de présence de symboles. Les données de ce module sont les boucles caractéristiques et les composantes texte provenant du bas niveau et l'ensemble des points extrémités des lignes de connexion déjà détectées. En sortie, nous obtenons une zone candidate si l'une des quatre régles de sélection est applicable sinon c'est un échec de la construction d'hypothèses. Nous verrons par la suite le traitement à effectuer dans ces deux cas.

5.3.2 Agrandissement de la zone candidate

La reconnaissance peut échouer sur une zone sélectionnée par le module de construction d'hypothèses et par conséquent, aucune confirmation de présence de

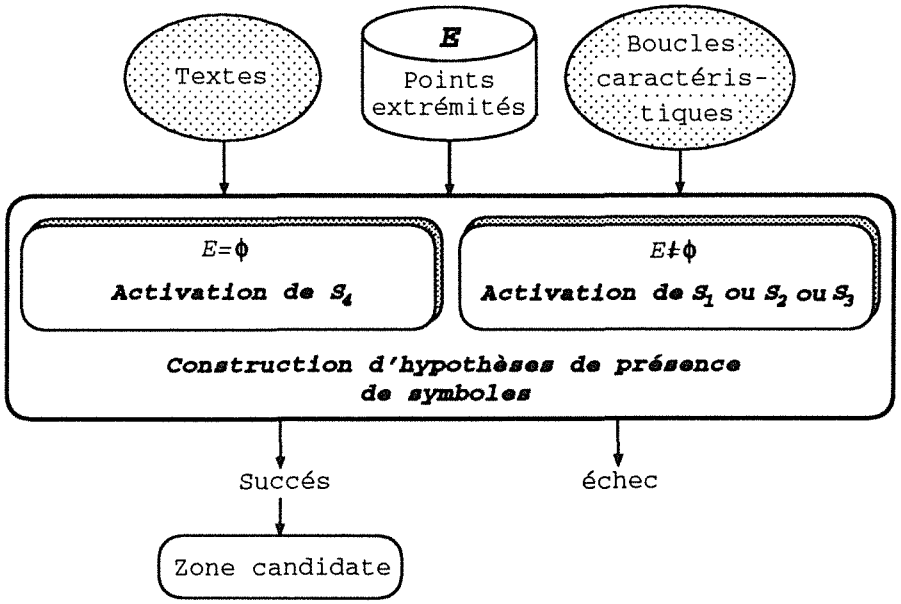


Figure 5.7. *Construction des hypothèses de présence de symboles .*

symbole dans cette zone n'est donnée. Ainsi, si la règle utilisée est (S_1) ou (S_4) alors la boucle caractéristique utilisée est considérée comme étant une *boucle problématique*, alors que si c'est la régle (S_2) ou (S_3) le point extrémité utilisé est considéré comme étant un *point problématique.*

Il semble judicieux de prévoir *une remise en cause* du résultat du module de construction d'hypothèses. La Figure 5.8.a montre un exemple qui illustre ce cas. En effet, si un symbole est constitué de plusieurs boucles caractéristiques, en utilisant l'une des règles S_1 ou S_4, nous risquons d'inclure dans la zone candidate uniquement une seule boucle caractéristique décomposant ainsi le symbole et rendant impossible sa reconnaissance. Un *agrandissement* de la zone sélectionnée semble alors être nécessaire.

Pour agrandir la zone candidate litigieuse, une étude du voisinage de la boucle caractéristique problématique est effectuée recherchant l'existence d'autres boucles caractéristiques qui lui sont connexes. La nouvelle zone candidate à sélectionner est alors définie comme étant le rectangle qui contient toutes les boucles caractéristiques connexes trouvées. Ainsi dans l'exemple de la Figure 5.8.a un agrandissement de la zone fournit une zone incluant les deux boucles caractéristiques constituant la résistance variable, symbole à reconnaître (Figure 5.8.b).

De même pour la règle (S_2), si le symbole est plus grand que la moyenne des tailles des boucles caractéristiques du schéma, un agrandissement est alors néces-

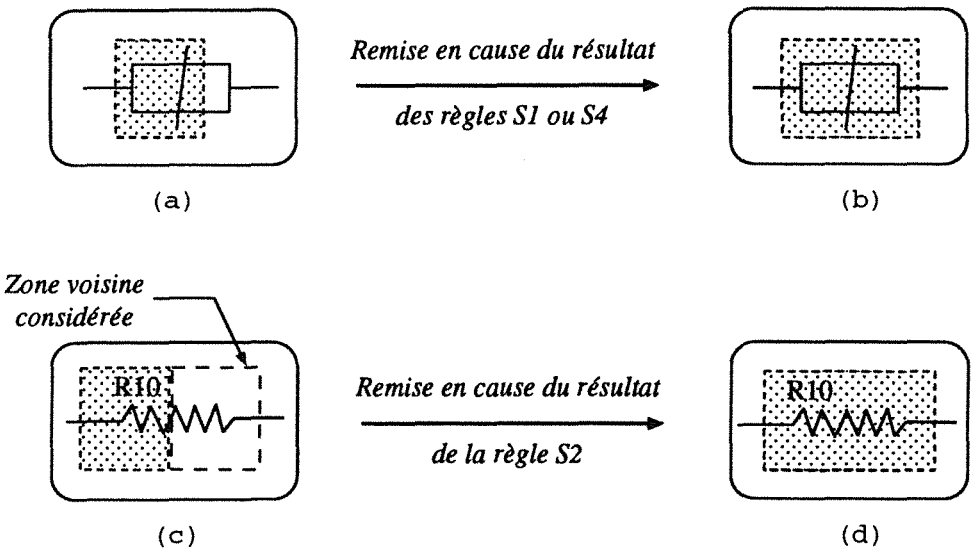


Figure 5.8. *Agrandissement de la zone candidate.*

saire. L'agrandissement est réalisée par une étude de la zone connexe à la zone sélectionnée. Si elle contient du texte ou un ensemble de segments, nous définissons une nouvelle zone candidate comme étant le rectangle englobant l'ancienne zone ainsi que la zone voisine. Par exemple dans la Figure 5.8.c, un agrandissement de la zone candidate fournit une zone incluant les deux parties constituant la résistance (Figure 5.8.d).

Si après agrandissement, aucun symbole n'a été reconnu, la boucle problématique ou le point problématique reste donc problématique et aucun nouveau agrandissement n'est déclenché.

Dans le cas où la zone a été sélectionnée par la règle ($\boldsymbol{S}_3$), aucun agrandissement n'est réalisé puisque dans ce cas l'hypothèse de présence de symbole n'a pas été appuyée par aucune information autre que les points extrémités et le voisinage du point extrémité a été étudié dès la sélection. Ainsi, les points problématiques générés par une remise en cause de la régle (S_3) le restent.

Nous venons de présenter comment réaliser l'agrandissement, il reste à présent, à décider le moment d'activation de l'opération d'agrandissement. Deux choix sont possibles :

1. appliquer l'agrandissement dès la remise en cause d'une règle de sélection,
2. appliquer l'agrandissement à la fin du traitement du schéma et donc après utilisation de tous les points extrémités et toutes les boucles caractéristiques.

Remarquons qu'une boucle problématique peut être détectée comme étant une fausse boucle au cours d'une étape ultérieure de l'interprétation. De ce fait, si la boucle problématique à étudier est une fausse boucle non encore détectée, son agrandissement immédiat en respectant le premier choix ne sert à rien. C'est pour cette raison, que nous préférons le choix 2 pour n'appliquer l'agrandissement sur l'ensemble des boucles problématiques qu'une fois les étapes d'interprétation épuisées.

En ce qui concerne les points problématiques générés par la remise en cause de la règle (S_2) de sélection de zone candidate, les deux choix pour l'application de l'agrandissement sont équivalentes. Tout de même, nous avons choisi d'appliquer l'agrandissement au même moment que pour les boucles problématiques pour homogénéiser le traitement d'agrandissement.

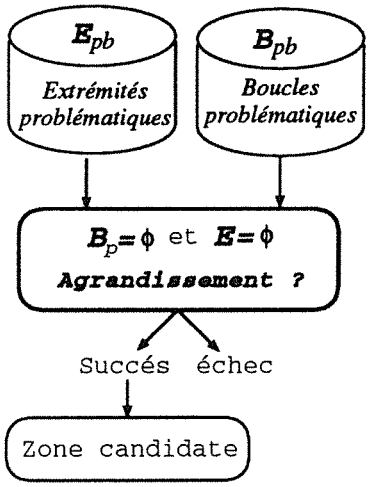


Figure 5.9. *Conditions d'applications de l'agrandissement.*

La Figure 5.9 montre les données et les sorties de cette opération d'agrandissement ainsi que les conditions d'application qui sont d'une part l'ensemble E des points extrémités est vide et d'autre part l'ensemble B_p des boucles caractéristiques *pertinentes* est vide. Une boucle pertinente est une boucle qui n'est actuellement ni fausse ni problématique.

5.4 Détection des lignes de connexion

Afin d'utiliser l'information points extrémités dès l'activation du système, nous souhaitons initialiser l'ensemble des lignes de connexion (cf § 5.4.1). A part le module d'initialisation, l'ensemble des lignes de connexion est enrichi en cours du traitement de trois manières (Figure 5.10) :

- *suivi à partir des points de connexion*: ainsi l'enrichissement est réalisé par le résultat du module de reconnaissance et les lignes qui relient le symbole reconnu au schéma sont des lignes de connexion.
- *suivi à partir des points de branchement*: les points de branchement d'une ligne de connexion L permettent la détection des lignes de connexion intersectant cette ligne L en ces points.
- *suivi à partir des points extrémités*: dans le cas où un point extrémité d'une ligne de connexion déjà détectée indique simplement un changement de direction de cette ligne, ce point extrémité est alors en même tenps un point extrémité ou point de branchement d'une autre ligne de connexion. Un suivi de cette nouvelle ligne de connexion à partir de ce point est nécessaire. Ceci est activée quand aucune des règles de sélection (S_1), (S_2) et (S_3) n'est applicable à ces points extrémités.

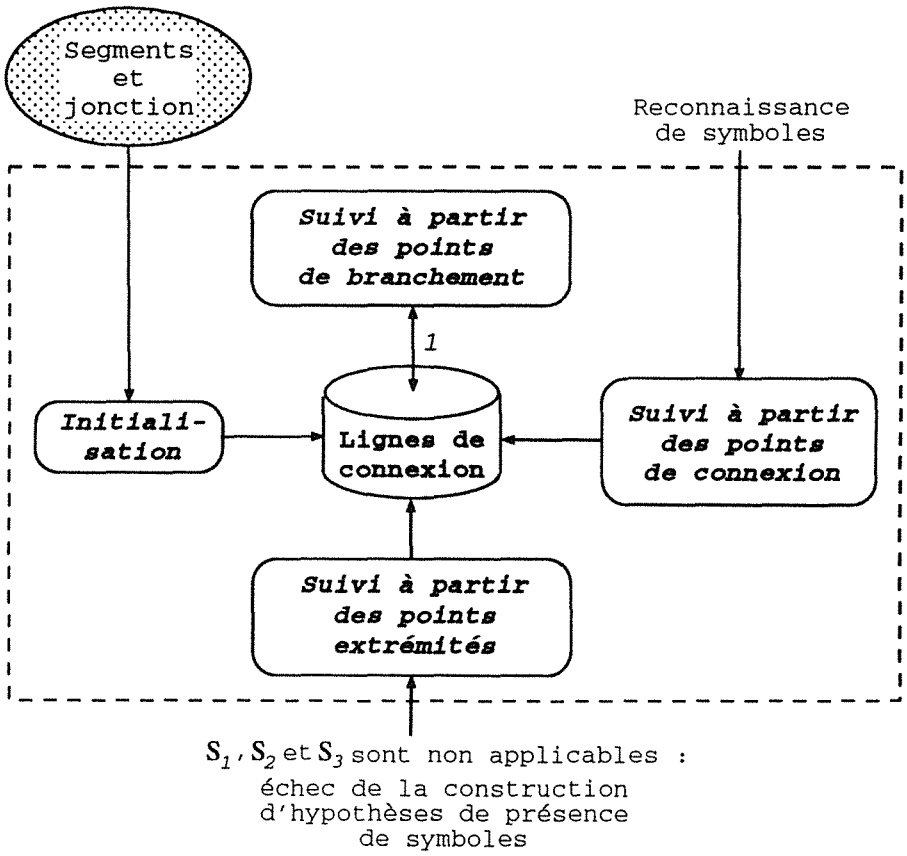


Figure 5.10. *Architecture détaillée du module de détection des lignes de connexion.*

La flèche 1 dans la Figure 5.10 signifie que tant qu'il y a de nouvelles lignes de connexion détectées, un enrichissement à partir de leurs points de branchement est activé.

Remarque soulignons que la règle (S_4) ne peut être déclenchée que si aucune nouvelle ligne de connexion n'a pu être détectée et donc aucun nouveau point extrémité n'a pu être généré. Ceci implique que même si les règles (S_1), (S_2) et (S_3) sont non applicables, le suivi à partir des points extrémités est plus prioritaire que l'activation de la règle (S_4) puisque ce suivi permet la détection de nouvelles lignes de connexion.

La Figure 5.11 montre les conditions d'activation de la détection de nouvelles lignes de connexion par suivi à partir des points particuliers des lignes de connexion déjà détectées.

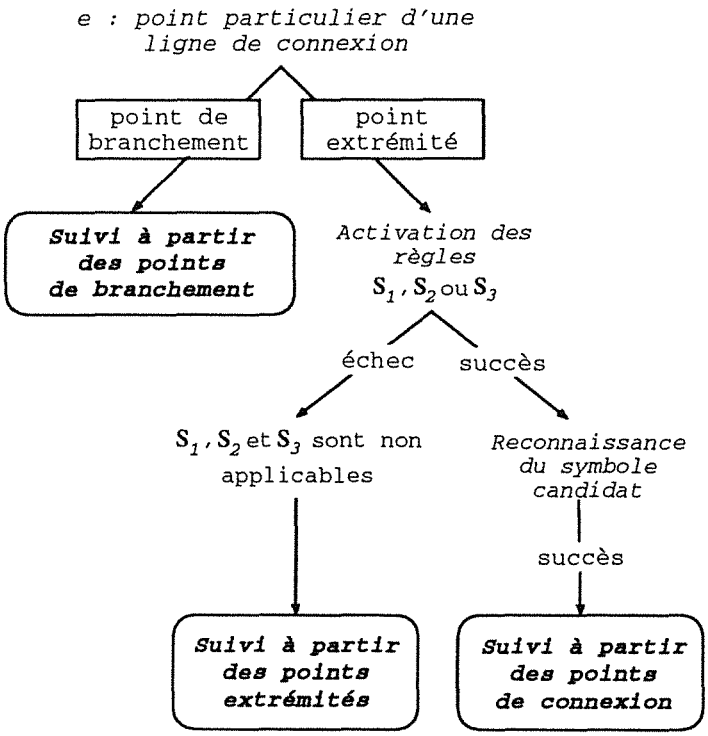


Figure 5.11. *Conditions d'activation de la détection des lignes de connexion.*

Tout changement affectant l'ensemble des lignes de connexion provoque nécessairement une mise à jour de l'ensemble des points extrémités. Dans cette section, nous montrons la gestion nécessaire pour tenir à jour l'information lignes de connexion et points extrémités.

Remarque dès qu'une nouvelle ligne de connexion est détectée, la détermination des fausses boucles est activée afin de vérifier si cette nouvelle ligne borde une boucle caractéristique en quel cas cette boucle est alors détectée fausse.

5.4.1 Initialisation de l'ensemble des lignes de connexion

Comme nous l'avons vu dans le chapitre 2, il est généralement admis dans les différents travaux que toutes les lignes verticales et horizontales de longueur supérieure à un seuil[2] donné sont des lignes de connexion. Cette hypothèse sur les directions des lignes de connexion sous-entend une numérisation droite du schéma par le scanner ou un redressement de l'image avant le traitement du schéma. Ceci constitue une restriction à l'application de notre système d'interprétation puisque le module de reconnaissance que nous avons développé est invariant par rapport à la rotation. En plus, avec l'hypothèse faite sur les longueurs, les petites lignes de connexion joignant deux symboles très proches ne seront jamais détectées.

Pour notre part, nous proposons un processus d'initialisation de l'ensemble des lignes de connexion qui ne fixe ni les directions de ces lignes, ni à l'avance la valeur du seuil pour leurs longueurs. L'idée consiste à étudier la répartition de la longueur de toutes les lignes du schéma au lieu d'initialiser l'ensemble des lignes de connexion de façon brutale par les lignes les plus longues du schéma. Cecipermet d'éviter le fait d'initialiser l'ensemble des lignes de connexion par des lignes qui ne le sont pas comme dans le cas d'un schéma représentant un symbole unique, ou encore le cas où les lignes de connexion ne sont pas les plus longues et ont une taille comparable à la taille des segments constituant les symboles. L'étude de la répartition des longueurs permet de vérifier s'il existe dans le schéma des lignes qui sont assez longues par rapport à celles du reste du schéma. L'idée consiste donc à trier les longueurs des lignes de connexion par ordre croissant puis à vérifier si les plus grandes valeurs sont éloignées des autres. Pour prendre plus de précaution, nous trions uniquement les lignes du schéma qui ne possèdent pas une partie appartenant à une boucle caractéristique. En effet, s'il existe une ligne parmi les plus grandes lignes de connexion trouvées qui appartient à une boucle, cette ligne risque d'appartenir à un symbole.

Après la réalisation du tri des lignes du schéma n'appartenant pas à des boucles, un partitionnement des lignes en deux classes est effectué. La classe des lignes les plus longues est la classe des lignes de connexion à l'aide duquel nous initialisons l'ensemble des lignes de connexion. Soulignons qu'aucun seuil n'est fixé pour la longueur des lignes de connexion, puisque à partir des modules de suivi, nous détectons les lignes de connexion ayant une longueur inférieure à la plus petite longueur de la classe obtenue par partitionnement.

Dans le cas où aucun partitionnement n'est possible, aucune initialisation n'est donc réalisée et l'ensemble des lignes de connexion reste vide. Le système est donc activé par le module de construction d'hypothèse de présence de symbole par l'application de la règle de sélection de la zone candidate ($\boldsymbol{S}_4$) (§ 5.3.1) qui utilise

2. Fixer un seuil à la longueur n'est pas très approprié et dépend du schéma traité.

uniquement les boucles caractéristiques du schéma.

Comme nous l'avons déjà dit l'initialisation de l'ensemble des lignes de connexion implique l'initialisation de l'ensemble des points extrémités. Par conséquent, le module d'initialisation opère en fait sur deux structures : l'ensemble L des lignes de connexion et l'ensemble E de leurs points extrémités.

5.4.2 Suivi à partir des points de connexion

L'enrichissement de l'ensemble des lignes de connexion se fait également par le module de reconnaissance à l'aide du suivi à partir des points de connexion. En effet, toute ligne contenant un point mis en correspondance avec un point de connexion du modèle est considérée comme étant une *ligne de connexion.* Ceci vient du fait que ces lignes relient le symbole reconnu au reste du schéma.

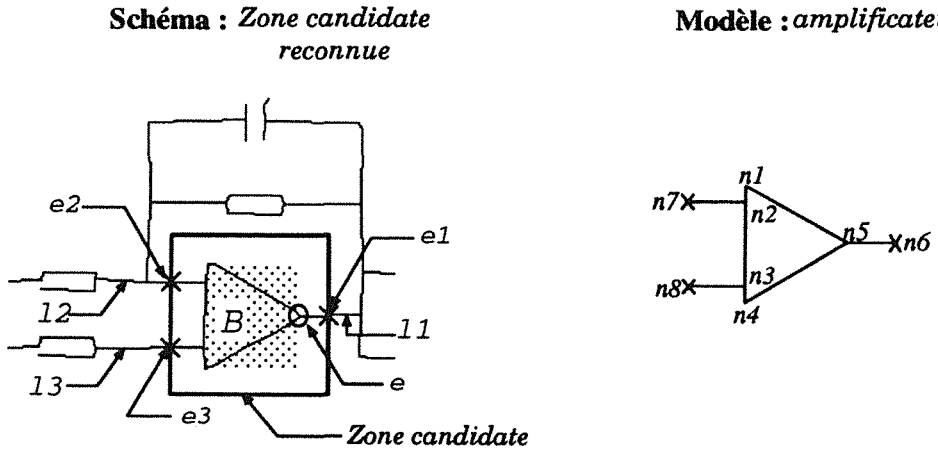


Figure 5.12. *Détection des lignes de connexion par suivi à partir des points de connexion.*

La Figure 5.12 montre un exemple qui illustre l'enrichissement de l'ensemble des lignes de connexion par le suivi à partir des points de connexion. Le rectangle en gras représente la zone candidate sélectionnée après application de la règle $(\boldsymbol{S}_1)$ (§ 5.3.1) utilisant le point extrémité e de la ligne de connexion[3] l_1 et la boucle caractéristique B. Une fois la reconnaissance effectuée, la zone candidate a été mise en correspondance avec l'amplificateur et les trois points e_1, e_2 et e_3 sont alors respectivement appariés avec n_6, n_7 et n_8, points de connexion du modèle. De part ce fait, les lignes l_2 et l_3 sont classées comme étant deux lignes de connexion puisqu'elles contiennent e_2 et e_3 reconnus comme étant des points de connexion. L'ensemble des lignes de connexion est alors enrichi par les lignes l_2 et l_3.

3. Déjà détectée auparavant comme étant une ligne de connexion.

L'enrichissement des lignes de connexion par le module de reconnaissance s'accompagne par un enrichissement de l'ensemble des points extrémités. Bien évidement pour chaque ligne détectée, uniquement le point extrémité n'appartenant pas au symbole est ajouté à l'ensemble E des points extrémités. Par exemple, l'extrémité de la lignes l_2 mis en correspondance avec le sommet n_2 n'est pas ajouté à l'ensemble E. La fonction permettant l'enrichissement des lignes de connexion admet comme données l'ensemble L des lignes de connexion, l'ensemble E des points extrémités et le symbole reconnu *symb*. Cette fonction se présente comme suit :

Fonction *Suivi_pt_connexion*$(L, E, symb)$
$L' = ligne\_pt\_connexion(symb)$ /* *symb* symbole reconnu */
$L' = L' \setminus (L' \cap L)$ /* élimination de L' les lignes de connexion appartenant déjà l'ensemble L */
Si $L' \neq \emptyset$ alors
$L = L \cup L'$
$E' = Extremite\_externe(L')$
$E = E \cup E'$
Fin Fonction

La fonction $ligne\_pt\_connexion(s)$ retourne les nouvelles lignes de connexion détectées à partir des points de connexion du symbole reconnu s. La fonction $Extremite\_externe(L')$ permet de déterminer pour chaque ligne le point extrémité n'appartenant pas au symbole reconnu.

5.4.3 Suivi à partir des points de branchement

Comme les lignes de connexion peuvent s'intersecter en des points différents de leurs extrémités, un suivi à partir des points de branchement permet de détecter de nouvelles lignes de connexion. Par conséquent, un enrichissement par toutes ces nouvelles lignes de connexion peut être effectué. Pour rendre cet enrichissement possible, il faut déterminer tous les points de branchement appartenant à la ligne de connexion courante.

La Figure 5.13 montre un exemple de détection de lignes de connexion par un suivi à partir des points de branchement. La ligne de connexion l possède deux points de branchement b_1 et b_2. Un suivi à partir de ces deux points de branchement fournit les deux nouvelles lignes de connexion l_1 et l_2. Par conséquent, ces deux lignes de connexion sont automatiquement ajoutées à l'ensemble des lignes de connexion.

Cette étape de détection de lignes de connexion est activée dès que l'ensemble des lignes de connexion est enrichi. Comme, la détection des lignes de connexion à partir des points de branchement affecte l'ensemble des lignes de connexion, le suivi

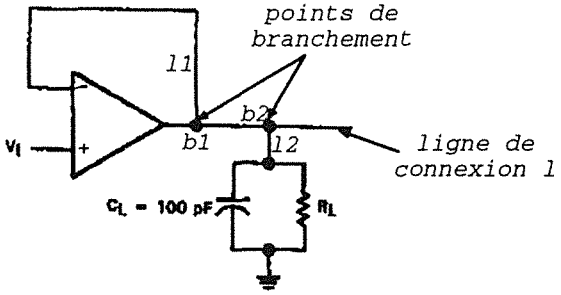


Figure 5.13. *Détection des lignes de connexion par suivi à partir des points de branchement.*

à partir des points de branchement est alors obligatoirement récursif. En effet, le suivi à partir des points de branchement est effectué jusqu'à ce qu'aucune nouvelle ligne ne puisse être engendrée et donc aucun nouveau point de branchement n'est détecté.

L'algorithme qui réalise la détection de nouvelles lignes de connexion à partir des points de branchement, se décompose donc en deux étapes :

- *première étape :* déterminer pour chaque nouvelle ligne tous ses points de branchement.

- *deuxième étape :* suivre les lignes passant par ces points de branchement afin de détecter de nouvelles lignes de connexion.

Notons que l'existence d'un point de branchement dans une ligne de connexion n'implique pas forcément la détection d'une nouvelle ligne de connexion dans notre système. En effet, dans le cas où le point de branchement appartient déjà à l'ensemble des points extrémités, la ligne qui lui est associée appartient donc déjà à l'ensemble des lignes de connexion. Dans de telles conditions, le suivi à partir des points de branchement se réduit à une simple mise à jour de l'ensemble des points extrémités duquel nous supprimons ce point de branchement. Par contre, dans le cas où le point de branchement n'appartient pas à l'ensemble des points extrémités, un enrichissement de l'ensemble des lignes de connexion doit être effectué.

A présent, présentons cette fonction de détection des lignes de connexion par un suivi à partir des points de branchement. Pour pouvoir décrire un tel algorithme, nous avons besoin de différencier les nouvelles lignes de connexion dont la détermination des points de branchement n'est pas encore effectuée de celles déjà traitées. Ainsi, en plus de l'ensemble des points extrémités E, nous définissons deux ensembles de lignes de connexion : le premier contenant les lignes de connexion *déjà traitées* et

noté L_t et le deuxième contenant les *nouvelles lignes* de connexion détectées et noté par L_n.

Le triplet (L_t, L_n, E) constitue les données de la fonction *Suivi_pt_branch* qui assure la détection de nouvelles lignes de connexion par un suivi à partir des points de branchement des lignes de connexion déjà détectées.

Fonction *Suivi_pt_branch*(L_t, L_n, E)
$B = \emptyset$ /* Ensemble des points de branchement */
Suivi_pt_branch_rec(L_t, L_n, E, B) /* réalise récursivement le suivi */
Fin Fonction

Fonction *Suivi_pt_branch_rec*(L_t, L_n, E, B)
Si $(B = \emptyset$ *et* $L_n = \emptyset)$ alors
fin suivi à partir des points de branchement
Sinon
Si $L_n = \emptyset$ alors /* $B \neq \emptyset$, traitement des points de branchement de B */
Prendre_une_boucle b *de* B, $B = B - \{b\}$
Si $b \in E$ alors $E = E - \{b\}$
Sinon /* le point de branchement b n'est pas un point extrémité */
$l' = Detect\_branch(b)$
$L_n = L_n \cup \{l'\}$
$E' = Pt\_extremite\_diff\_branch(l', b)$
$E = E \cup E'$
Fsi
Sinon /* B est vide, détermination des points de branchement */
Prendre_une_ligne l *de* L_n
$B = Pt\_branch(l)$
$L_n = L_n - \{l\}$
$L_t = L_t \cup \{l\}$
Fsi
Suivi_pt_branch_rec(L_t, L_n, E, B)
Fsi
Fin Fonction

La fonction $Pt\_branch(l)$ effectue la détermination des points de branchement d'une ligne de connexion non encore traitée. Cette ligne passe alors dans l'ensemble L_t des lignes de connexion traitées. La fonction $Detect\_branch(b)$ suit la nouvelle ligne de connexion à partir du point de branchement b. Enfin la fonction $Pt\_extremite\_diff\_branch(l, b)$ détermine les points extrémités de l différents du point de branchement b. La Figure 5.14 montre le cas où le point de branchement

peut être une extrémité et celui où il ne l'est pas. En effet, le premier cas correspond au point de branchement b_1 qui est en même temps une extrémité de la nouvelle ligne détectée l_1 ; dans ce cas un seul nouveau point extrémité est trouvé. Par contre le deuxième cas correspond au point de branchement b_2 qui appartient à la nouvelle ligne détectée l_2 sans être l'une de ses extrémités ; dans ce cas deux nouveaux points extrémités sont détectés.

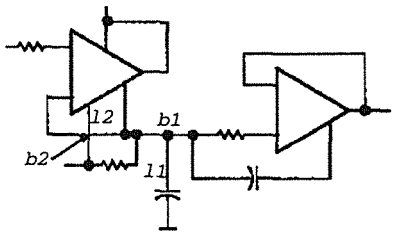


Figure 5.14. *Mise à jour de l'ensemble des points extrémités.*

La détection des lignes de connexion à partir des points de branchement est terminée lorsqu'il n'y a plus ni de nouvelles lignes à traiter, ni de points de branchement.

5.4.4 Suivi à partir des points extrémités

Le fait que toutes les règles (S_1), (S_2) et (S_3), utilisant les points extrémités des lignes de connexion, soient non applicables, signifie que les points extrémités courants n'admettent pas dans leur voisinage de symboles. Ceci est tout à fait possible puisque, à l'extrémité des lignes de connexion et bien qu'il y ait une forte probabilité de présence de symbole nous pouvons avoir :

- juste un changement de directions des lignes de connexion et par conséquent, toutes les autres lignes intersectant la ligne de connexion courante à ce point extrémité sont également des lignes de connexion,
- ou bien des *points finaux.*

La Figure 5.15 illustre ce fait ; en effet, prenons par exemple l'extrémité e_3 de la ligne de connexion l_2 déjà détectée, nous constatons qu'il y a un simple changement de direction de lignes de connexion et qu'à cette extrémité aucun symbole n'existe. Par contre le point extrémité e_2 de la ligne de connexion l_1 est un *point final* du schéma et ne nécessite aucun traitement.

Par conséquent, dans le cas où les règles (S_1), (S_2) et (S_3) sont non applicables, un suivi à partir des points extrémités à la recherche de nouvelles lignes de connexion s'avère nécessaire.

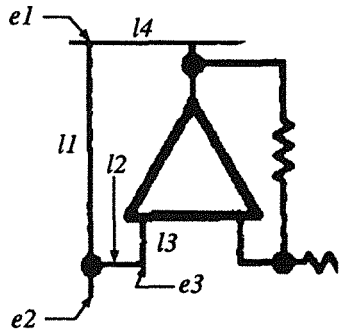


Figure 5.15. *Détection des lignes de connexion par suivi à partir des points extrémités.*

Ce suivi est assuré par la fonction *Suivi_pt_extremite* qui admet comme données l'ensemble L des lignes de connexion et l'ensemble E de leurs points extrémités.

Fonction *Suivi_pt_extremite*(L, E)
$E_n = \emptyset$ /* Ensemble des nouveaux points extrémités
détectés à partir des points de E */
tant que $E \neq \emptyset$ faire
Prendre_une_extremie e *de* E ; $E = E - \{e\}$
$L' = Detect\_ligne\_extremite(e)$
Si $L' \neq \emptyset$ alors /* nouvelles lignes de connexion ont été détectées */
$E' = Pt\_extremite\_diff\_extremite(L', e)$
$E_n = E_n \cup E'$
$L = L \cup L'$
Sinon /* $L' = \emptyset$ donc aucune nouvelle ligne n'a été détecté
à partir de e */
e *est un point final*
Fin tant que
$E = E_n$
Fin Fonction

La fonction $Detect\_ligne\_extremite(e)$ détecte les nouvelles lignes de connexion à partir du point extrémité e. La fonction $Pt\_extremite\_diff\_extremite(L', e)$ détermine les points extrémités, différents du point extrémité e, des lignes de connexion de L'.

Dans la Figure 5.15, supposons que nous avons $L = \{l_1, l_2\}$ et $E = \{e_1, e_2, e_3\}$, dans ce cas les trois règles (S_1), (S_2) et (S_3) de sélection de zones candidates sont non applicables et par conséquent, un suivi à partir des points extrémités est à effectuer.

Nous remarquons que pour e_2 aucune détection de nouvelles lignes de connexion n'est possible par contre pour l'extrémité e_3, une nouvelle ligne de connexion l_3 vient enrichir l'ensemble L et de même pour le point extrémité e_1, l_4 est détectée comme une nouvelle ligne de connexion.

5.5 Processus d'interprétation

Dans cette section, nous présentons le fonctionnement général du processus d'interprétation de notre système d'analyse du schéma et dont le rôle principal consiste à gérer les échanges entre tous les modules constituant le processus d'interprétation. En d'autres termes, ceci revient à intégrer toutes les fonctions constituant l'étape de détection des lignes de connexion et à instaurer le contrôle nous permettant de gérer les règles $(\boldsymbol{S}_1)$, $(\boldsymbol{S}_2)$, $(\boldsymbol{S}_3)$ et $(\boldsymbol{S}_4)$ de construction d'hypothèses de présence de symboles ainsi que l'agrandissement des zones pour résoudre les cas problématiques.

La Figure 5.16 décrit le fonctionnement du processus d'interprétation. Les données de ce processus sont fournit par le bas niveau et sont constituées par les segments, les points de jonctions, les composantes textes et l'ensemble des boucles caractéristiques. Si le processus d'initialisation réussit à détecter des lignes de connexion, l'ensemble E des points extrémités est alors affecté ; dans le cas contraire les deux ensembles L et E restent vide. Une fois le traitement d'initialisation fait, la sélection des zones candidates est effectuée en utilisant l'une des règles (S_1), (S_2), (S_3) et (S_4) en respectant l'ordre de priorité instauré sur ces règles et les conditions de leur applications. Si les règles (S_1), (S_2) et (S_3) sont non applicable alors un enrichissement à partir des points extrémités est activé. Si cette dernière étape ne permet pas la détection de nouvelles lignes de connexion, les points extrémités sont alors dits finaux[4] et ils sont supprimés de l'ensemble E. Par contre dans le cas où (S_3) est temporairement applicable, le point extrémité est considéré comme point problématique et aucun suivi à partir de ce point extrémité n'est effectué. Quand l'ensemble des points extrémités E est vide la sélection des zones candidates est alors effectuée par la règle (S_4). Cette condition d'utilisation de la règle (S_4) traduit la priorité des trois règles (S_1), (S_2) et (S_3) de construction d'hypothèses de présence de symboles par rapport à la règle (S_4). Une fois une zone candidate sélectionnée, le module de reconnaissance est activé sur cette zone avec la base de modèles adéquate ; base contenant les symboles avec boucles ou les autres symboles et ceci suivant la règle appliquée. Deux cas se présentent alors :

- soit le symbole est reconnu. L'hypothèse de présence de symbole est alors confirmée par le module de reconnaissance. Cette reconnaissance est alors sui-

4. Il ne contiennent pas de symboles à leur voisinage et ils n'indiquent pas changement de direction.

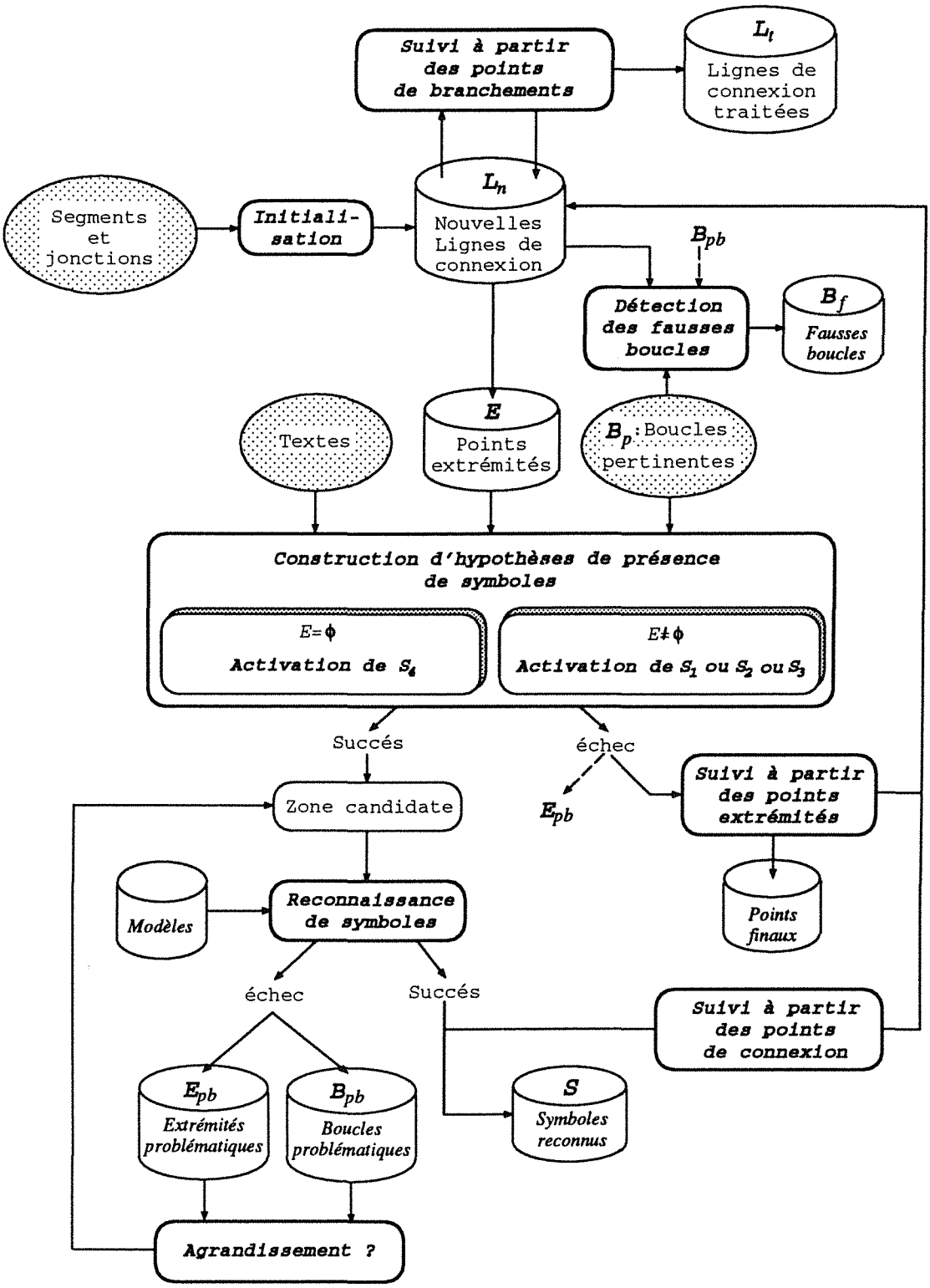


Figure 5.16. *Fonctionnement du processus d'interprétation de notre système.*

vit d'un enrichissement de l'ensemble des lignes de connexion et donc de l'enrichissement de l'ensemble des points de connexion,

- soit la reconnaissance échoue. Dans ce cas, nous sommes en présence d'un cas problématique. Si la règle (S_1) ou (S_4) qui est utilisée alors la boucle caractéristique est considérée comme une boucle problématique et elle est ajoutée à l'ensemble B_{pb}. Par contre dans le cas où c'est règle (S_2) ou (S_3) qui est utilisée alors le point extrémité est considéré comme un point problématique et il est ajouté à l'ensemble E_{pb}.

Dès qu'une nouvelle ligne de connexion est détectée, nous activons le module de détection de fausses boucles. Ce module vérifie si une partie de cette nouvelle ligne constitue le contour d'une boucle caractéristique problématique ou pas. Si c'est le cas, la boucle change d'état et passe dans l'ensemble B_f des fausses boucles. Une fois la détection des fausses boucles terminée, le suivi à partir des points de branchement de cette nouvelle ligne de connexion est alors activé. Bien entendu l'ensemble E des points extrémités est automatiquement mis à jour. Cette nouvelle ligne est alors mise dans l'ensemble L_t des lignes de connexion traitées.

Comme nous l'avons vu dans § 5.3.2, le traitement des cas problématiques par agrandissement est effectué quand d'une part les points extrémités ont été traités et qu'aucun nouveau point ne peut être engendré et que d'autres part toutes les boucles caractéristiques ont été traitées. Ceci se traduit par le fait que l'ensemble E des points extrémités est vide, que l'ensemble L_n des nouvelles lignes de connexion est aussi vide et que l'ensemble B_p des boucles caractéristiques *pertinentes*[5] est également vide.

La fin de l'interprétation survient quand :

- tous les points extrémités ont été traités : $E = \emptyset$,
- aucun nouveau point ne peut être engendré : $L_n = \emptyset$,
- toutes les boucles caractéristiques pertinentes ont été traitées : $B_p = \emptyset$,
- toutes les boucles caractéristiques problématiques ont été traitées par l'opération agrandissement : $B_{pb} = \emptyset$,
- les points extrémités problématiques, ne provenant pas d'une remise en cause de la règle (S_3), ont été traitées par l'opération agrandissement : $E_{pb} = \emptyset$.

Nous obtenons comme résultat final l'ensemble S contenant tous les symboles reconnus, l'ensemble L_t contenant toutes les lignes de connexion détectées, l'ensemble

5. Elle n'est ni problématique, ni reconnue, ni classée comme fausse boucle.

E^f_{pb} contenant les points extrémités problématiques qui demeurent problématiques même après agrandissement, l'ensemble B^f_{pb} contenant toutes les boucles caractéristiques problématiques qui restent définitivement problématiques même après agrandissement et l'ensemble B_f contenant toutes celles détectées comme fausses boucles (Figure 5.17).

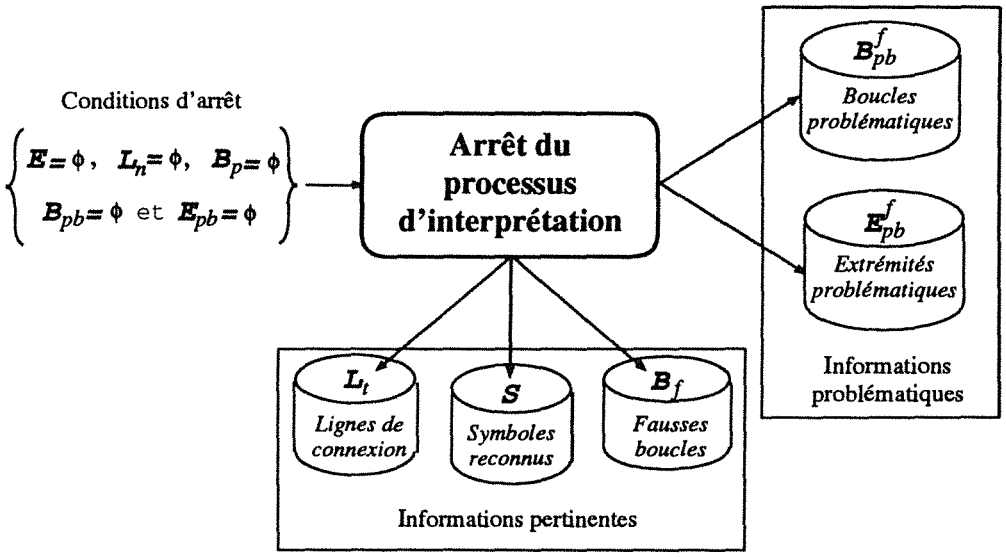


Figure 5.17. *Conditions d'arrêt du processus d'interprétation.*

Nous remarquons qu'à tout momemt, notre système est complètement défini par l'état de tous les ensembles que nous avons utilisé pour décrire son fonctionnement. Il nous a paru donc intéressant de le présenter en terme *de règles d'inférence* décrivant les transitions du système d'un état à un autre (cf annexe B).

Donnons à présent l'algorithme de la fonction $Interpretation(L_n, E, B_p, B_{pb}, E_{pb})$ qui réalise l'interprétation d'un schéma donné. Cette fonction travaille sur cinq ensembles $(L_n, E, B_p, B_{pb}, E_{pb})$ passés en paramètres qui sont les trois ensembles qui contrôlent l'arrêt de l'interprétation et de leur contenu dépend le traitement à faire à un instant donné. B_p est initialisé par toutes les boucles caractéristiques du schéma détectées par le processus du bas niveau. E_{pb}, B_{pb} sont initialisés à l'ensemble vide. Le processus d'interprétation est réalisé en deux étapes :

- *étape d'initialisation :* elle réalise l'initialisation de l'ensemble des lignes de connexion et est présentée dans la section 5.4.1. Ceci se traduit par l'initialisation des deux ensembles L_n et E. Si cette étape échoue, L_n et E sont initialisés à l'ensemble vide.

- *étape d'interpretation proprement dite :* elle est effectuée par la fonction suivante :

Fonction *Interpretation*($L_n, E, B_p, B_{pb}, E_{pb}$)
Si $L_n \neq \emptyset$ alors
Detect_fausse_boucle(L_n, B_p, B_f, B_{pb}) /* B_p et B_{pb} peuvent changer */
Suivi_pt_branch(L_t, L_n, E) /* L_n change, E peut changer */
Sinon /* on n'a pas de nouvelles lignes de connexion */
Si $E = \emptyset$ *et* $B_p = \emptyset$ *et* $B_{pb} = \emptyset$ *et* $E_{pb} = \emptyset$ alors
/* Fin de l'interprétation */
Arrêt de l'analyse du schéma
Sinon
Si $E = \emptyset$ *et* $B_p = \emptyset$ alors
Agrandissement($B_{pb}, E_{pb}, B^f_{pb}, E^f_{pb}$) /* B_{pb} ou E_{pb} change */
Sinon /* Création d'une zone candidate et sa reconnaissance */
Si $E = \emptyset$ alors
nouvelle_zone=Zone_cand(S_4)
Sinon /* L'ensemble des points extrémités est non vide */
nouvelle_zone=Zone_cand(S_1 *ou* S_2 *ou* S_3)
Si *pas(nouvelle_zone)* alors
Si (S_3) *est temporairemnt non applicable* alors
Mise_a_jour(E_{pb}) /* E et E_{pb} change */
Sinon
Suivi_pt_extremite(L_n, E) /* E change,
L_n peut changer */
Fsi
Fsi
Fsi
symb=Reconnaissance(nouvelle_zone,modèles)
Si *Reconnu(symb)* alors
Suivi_pt_connexion($L_n, E, symb$) /* L_n et E changent */
Enrichissement_symbole($S, symb$) /* B_p peut changer */
Sinon /* Symbole non reconnu */
Mise_a_jour(E_{pb}, B_{pb}) /* E change ou B_p change */
Fsi
Fsi
Fsi
Interpretation($L_n, E, B_p, B_{pb}, E_{pb}$)
Fin Fonction

Les fonctions *Suivi_pt_branch*, *Suivi_pt_extremite* et *Suivi_pt_connexion* sont les fonctions présentées dans la section 5.4. *Zone_cand* effectue la sélection de la zone candidate en appliquant l'une des règles de sélection donnée en paramètre tout

en respectant l'ordre de priorité défini sur ces règles (section 5.3.1). La fonction *Reconnaissance* effectue la reconnaissance d'un symbole candidat en utilisant une base de modèles. Ce module a fait l'objet d'une étude détaillée dans le chapitre 4. La fonction $Detect\_fausse\_boucle(B_p, L_n, B_f)$ détermine si parmi l'ensemble des boucles caractéristiques pertinentes, il existe des boucles qui comportent une ligne de connexion de l'ensemble L_n dans leur composition. Toutes les boucles vérifiant cette condition sont écartées de l'ensemble B_p et sont placées dans l'ensemble B_f des fausses boucles. L'agrandissement de la zone candidate sélectionnée est présenté dans la section 5.3.2 et est effectuée par la fonction $Agrandissement(B_{pb}, E_{pb}, B^f_{pb}, E^f_{pb})$. Les boucles caractéristiques ou les points extrémités qui restent problématiques même après agrandissement sont respectivement mis dans les ensembles B^f_{pb} et E^f_{pb}. La fonction $Enrichissement\_symbole(S, symb)$ ajoute tout simplement à l'ensemble S des symboles reconnus le nouveau symbole *symb*. De même, la fonction $Mise\_a\_jour(E_{pb}, B_{pb})$ met à jour l'ensemble B_{pb} dans le cas où la règle de sélection de zone candidate utilisée est (S_1) ou (S_4). Par contre si la règle utilisée est (S_2) ou (S_3), c'est l'ensemble des extrémités problématiques E_{pb} qui est enrichi.

5.6 Résultats d'exécution

Nous venons de détailler notre approche d'analyse contextuelle de schémas complets. Dans cette section, nous validons cette approche par différents résultats expérimentaux. Les deux premiers exemples seront détaillés afin de montrer les différentes étapes de l'analyse. Nous notons que la manière dont le système est activé est différentes pour ces deux exemples. Quant aux deux derniers exemples, nous présentons uniquement les résultats finaux obtenus.

La base de modèles utilisée pour nos expérimentations est donnée par la Figure 5.18. Elle contient douze symboles contenants des boucles et deux symboles sans boucles. Nous remarquons par la suite que les symboles reconnus n'ont pas toujours la même orientation et la même taille que les modèles.

Avant de présenter les différents résultats, nous expliquons la légende utilisée pour leurs affichage lors de l'analyse :

- *trait fin :* les lignes sont non encore reconnues,
- *trait fort :* les lignes sont détectées comme étant des lignes de connexion,
- *rectangle fin :* la zone candidate sélectionnée est non encore reconnue,
- *rectangle fort :* le symbole contenu dans cette zone est reconnu,
- *rectangle pointillé :* la boucle contenue dans ce rectangle est une boucle problématique.

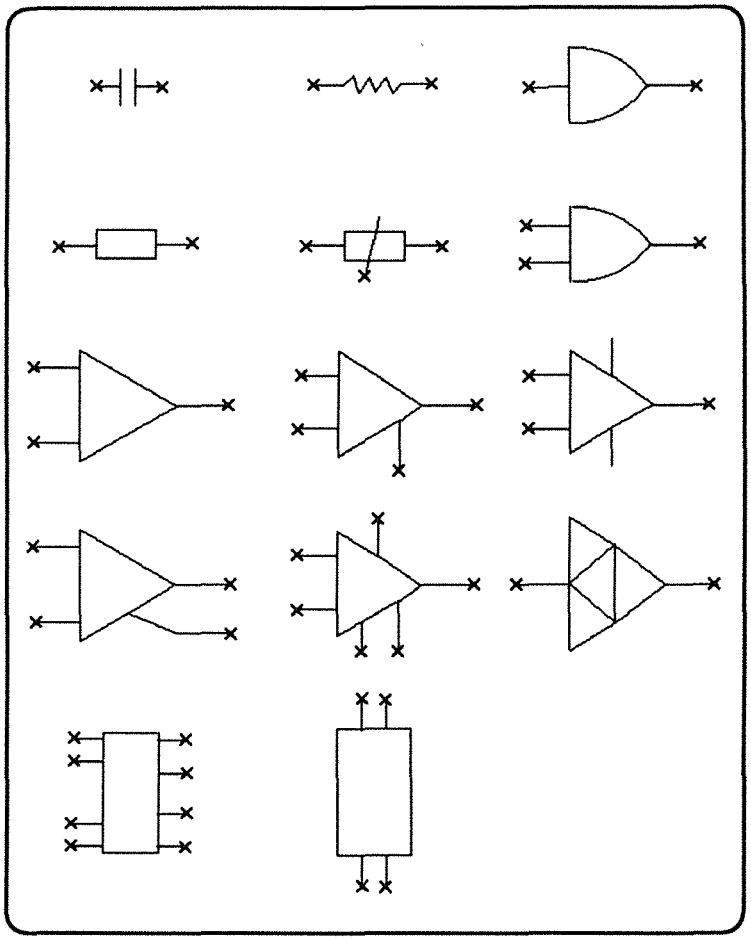


Figure 5.18. *Base de modèles utilisée pour l'expérimentation.*

Nous tenons à signaler que ces résultats constituent indirectement une validation du processus de reconnaissance.

5.6.1 Cas d'activation de notre système par initialisation de l'ensemble des lignes de connexion

La Figure 5.19 montre le schéma à analyser ainsi que l'ensemble de segments obtenus après application du processus du bas niveau.

La Figure 5.20.a montre le résultat de l'initialisation de l'ensemble des lignes de connexion ainsi que la première sélection de zone candidate. En effet, l'ensemble des lignes de connexion a été initialisé à la ligne L. Cette initialisation a activé le suivi à partir du point de branchement b qui a enrichi les lignes de connexion par les lignes $L1$.

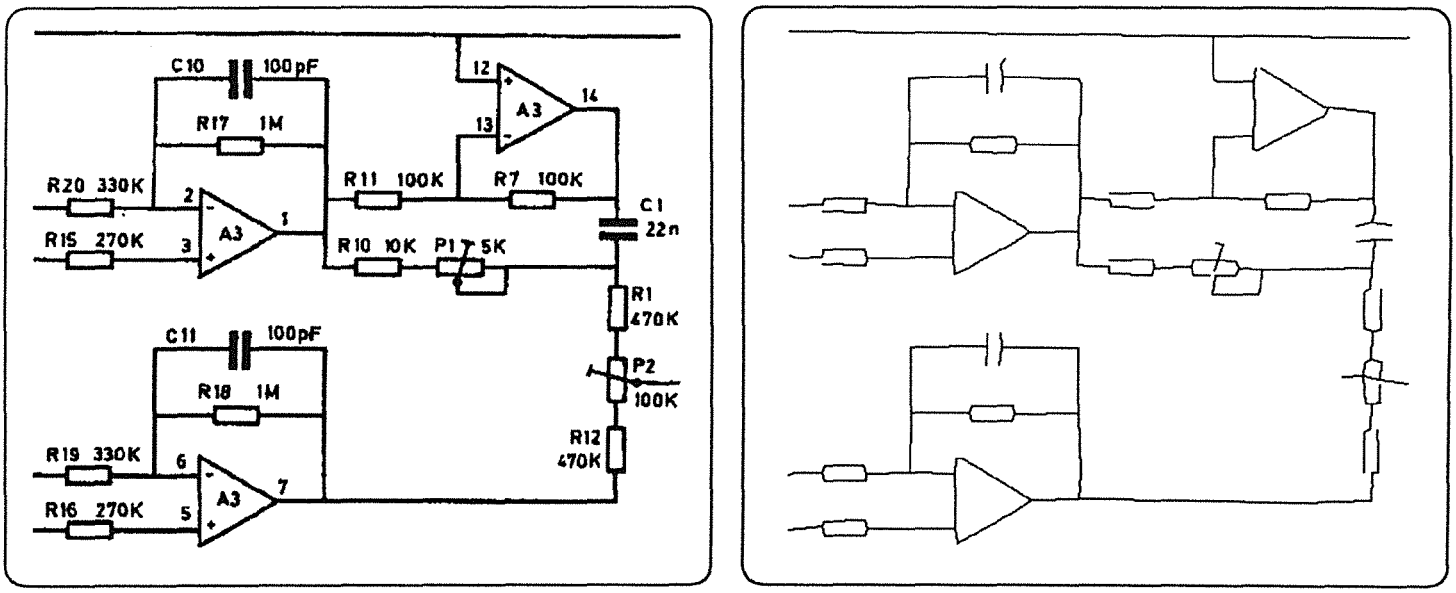


Figure 5.19. *(a) Image originale, (b) Ensemble de segments obtenus après application du processus du bas niveau sur (a).*

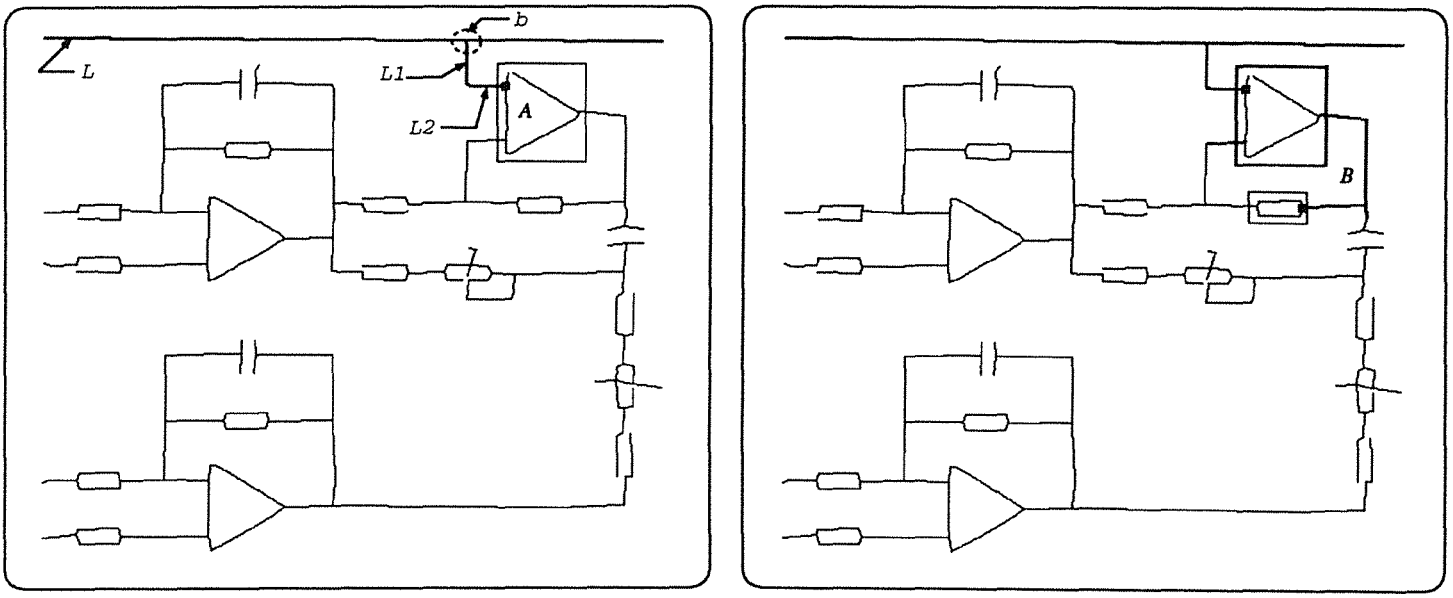


Figure 5.20. *Initialisation de l'ensemble des lignes de connexion et sélection de zone candidate en utilisant la règle* (S_1).

A cette étape de l'analyse, aucune des règles de sélection ne peut être appliquée et par conséquent, un suivi à partir du point extrémité de la lignes de connexion $L1$ permet alors la détection d'une nouvelle ligne de connexion $L2$. Cette détection a permis la sélection d'une zone candidate à reconnaître (rectangle de la Figure 5.20.a). Cette construction d'hypothèses de présence de symboles est effectuée par l'application de la règle (S_1) de sélection en utilisant le point extrémité matérialisé par le petit rectangle noir et la boucle carcatéristique A. Après application de la reconnaissance, nous constatons que l'amplificateur contenu dans cette zone a été reconnu et de nouvelles lignes de connexion sont détéctées (Figure 5.20.b). Ceci a permis une deuxième application de la régle (S_1) impliquant la selection d'une nouvelle zone à reconnaître.

Remarquons que la boucle B est détectée comme étant une fausse boucle puisque ces contours sont reconnus comme étant des lignes de connexion. Ceci valide notre choix d'utilisation des liens contextuels afin de minimiser le nombre de fausses boucles traitées par la reconnaissance. Mais nous n'éliminons pas complètement ce traitement des fausses boucles. En effet, la Figure 5.21.a montre la nouvelle zone sélectionnée (rectangle en trait fin) en utilisant le point extrémité e et la fausse boucle C. Malgré cette utilisation de fausse boucle, une résistance est reconnue dans cette zone candidate et de nouvelles lignes de connexion sont détectées (Figure 5.21.a). Ceci a permis de sélectionner une nouvelle zone à reconnaître contenant un amplificateur (rectangle fin de la Figure 5.21.b).

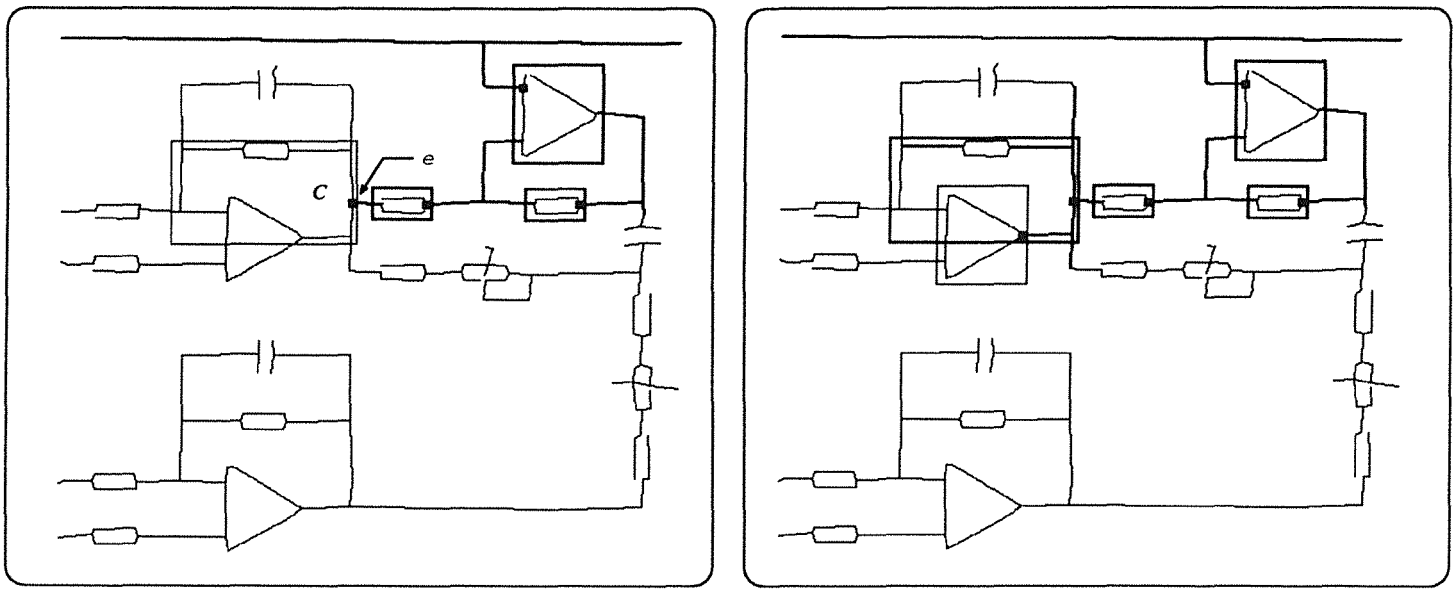


Figure 5.21. *Sélection d'une zone candidate en utilisant une fausse boucle.*

Dans la Figure 5.22, nous remarquons que cette amplificateur est reconnu. Ainsi,

cette étape de l'analyse illustre l'utilisation de la règle (S_2) qui utilise l'information texte. En effet, nous constatons qu'une nouvelle zone à reconnaître contenant la capacité est sélectionnée. Ce symbole est reconnu (rectangle devenu en trait fort dans la Figure 5.23).

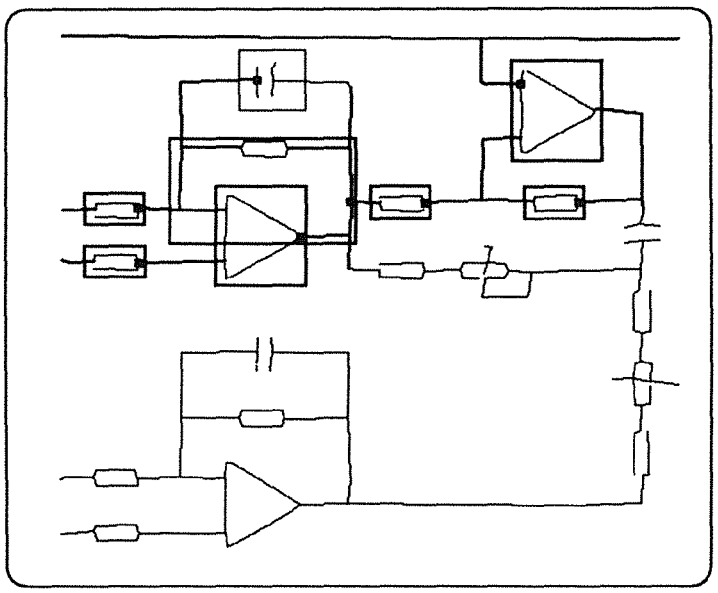


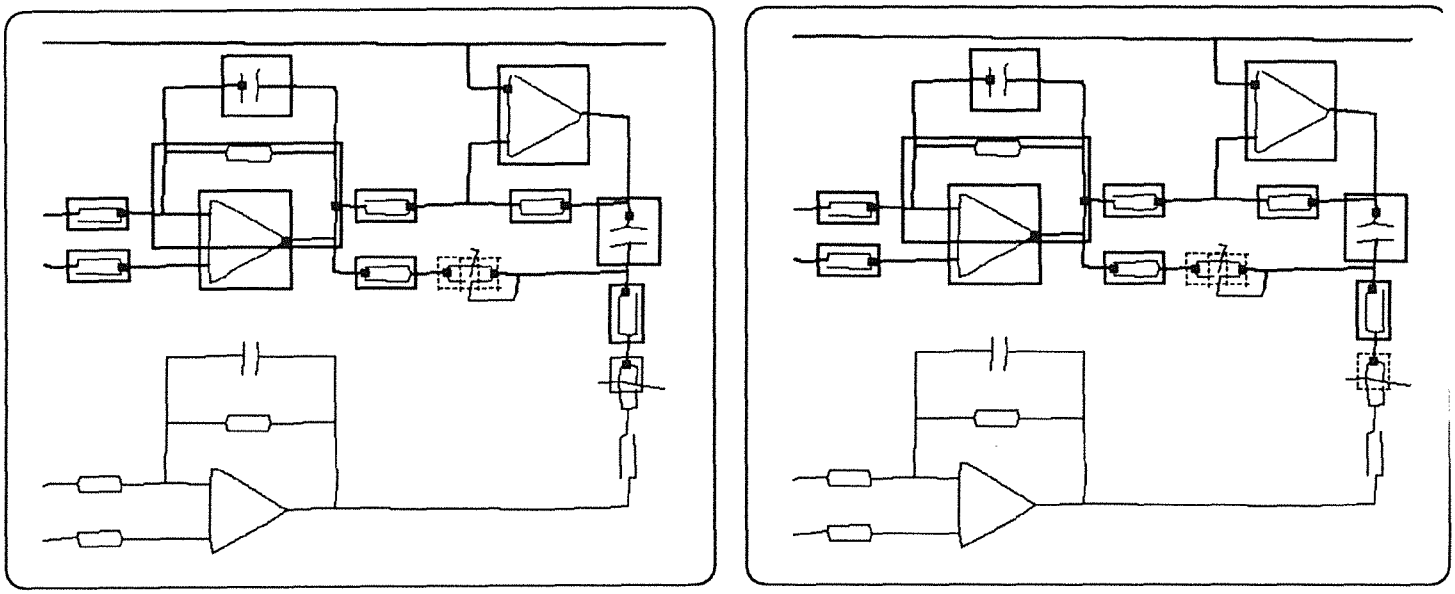
Figure 5.22. *Utilisation de la règle (S_2) pour la sélection de zone candidate.*

La boucle utilisée pour la sélection de la zone à reconnaître de la Figure 5.23.a est transformée en une boucle problématique (rectangle pointillé de la Figure 5.23.b). Ceci provient du fait que la zone sélectionnée ne contient qu'une partie du symbole.

Nous remarquons qu'à ce niveau de l'analyse aucune nouvelle lignes de connexion ne peut être détectée. Par conséquent le reste du schéma ne peut être atteint par un simple suivi des lignes de connexion. Ce qui explique l'application de la règle (S_4) de sélection de zone candidate en utilisant uniquement l'information boucle caractéristique (Figure 5.24).

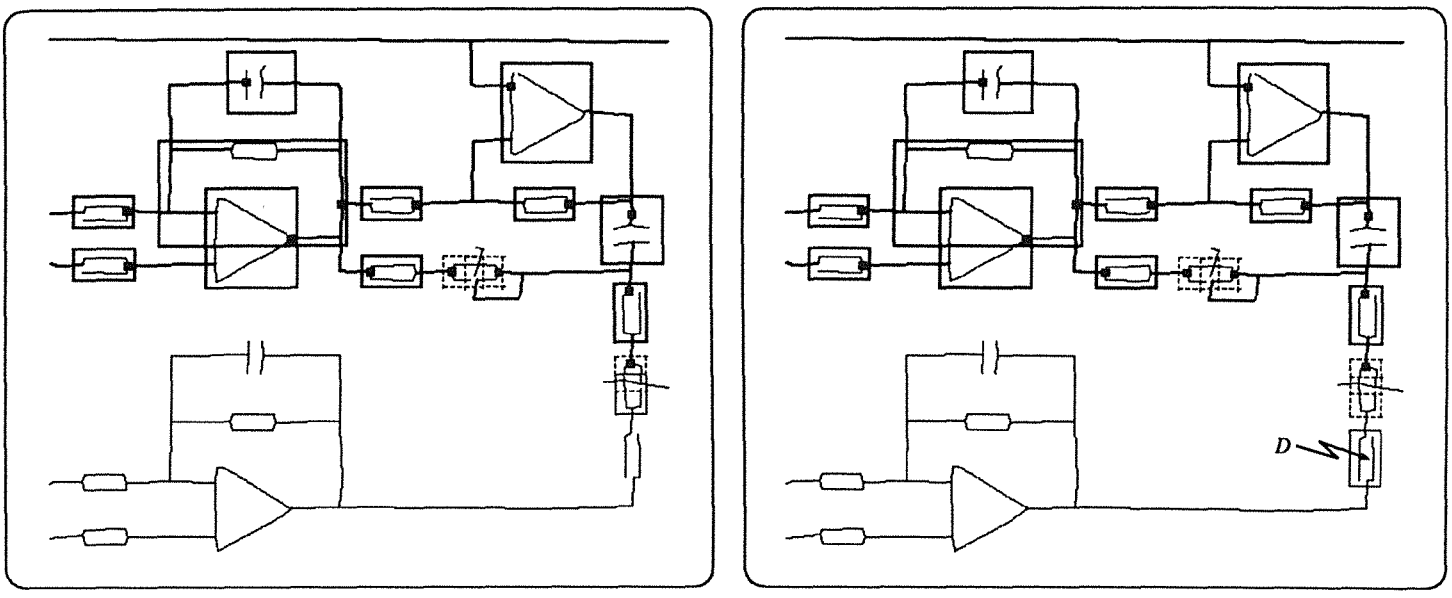
Une première application de cette règle permet la selection de la deuxième partie de la resistance variable (Figure 5.24.a) qui se transforme en une boucle problématique (Figure 5.24.b après échec de la reconnaissance sur cette zone). Cela implique, une deuxième application de la règle (S_4) de sélection en utilisant la boucle caractéristique D. Après reconnaissance de ce symbole (Figure 5.25), nous remarquons que l'ensemble des lignes de connexion est de nouveau initialisé. Par conséquent les règles les plus prioritaires fondées sur l'utilisation des points extrémités sont de nouveau applicables.

La Figure 5.26 montre l'état du schéma quand tous les points extrémités et toutes les boucles caractéristiques sont utilisés. Nous constatons que toutes les lignes de connexion sont détectées ainsi que tous les symboles à l'exception des deux résis-



(a) (b)

Figure 5.23. *Cas d'échec de la reconnaissance en engendrant une boucle problématique.*



(a) (b)

Figure 5.24. *Utilisation de la règle* (S_4) *de sélection de zone candidate.*

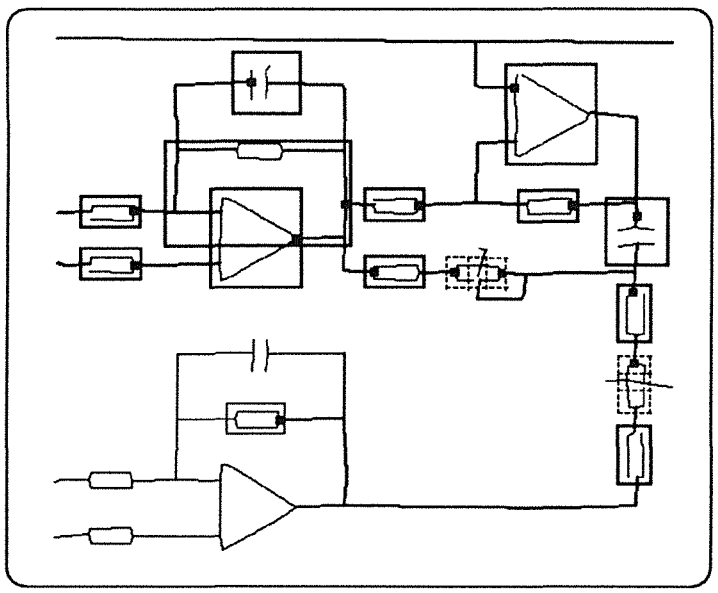


Figure 5.25. *Enrichissement de l'ensemble des lignes de connexion indirectement par la règle (S_4) de sélection de zone candidate.*

tances variables. A ce niveau de l'analyse, elles sont constituées chacune de deux boucles problématiques. Une étude du voisinage de ces boucles est alors effectuée dans le but d'activer l'opération agrandissement.

Ainsi, la Figure 5.27.a montre les deux boucles problématiques détectées comme étant connexes (les deux rectangles pointillés sont éliminés). Après agrandissement, de la zone étudiée, une nouvelle zone est sélectionnée contenant la totalité du symbole (Figure 5.27.b). Ceci a permis la reconnaissance du symbole (Figure 5.28.a, le rectangle fin dans la Figure 5.27.b est devenu fort dans la Figure 5.28.a). La deuxième résistance variable est aussi sélectionnée après étude des deux boucles problématiques la constituant (Figure 5.28).

Nous montrons le résultat final dans la Figure 5.29.a. Nous pouvons constater ainsi que tous les symboles ont été sélectionnés et reconnus et que toutes les lignes de connexion ont été détectées. La Figure 5.29.b montre qu'une seule fausse boucle sur les quatre fausses boucles du schéma a été utilisée (boucle numéro **3**). Néanmoins, elle est immédiatement reconnu comme étant une fausse boucle puisque la zone sélectionnée a permis la reconnaissance de la résistance.

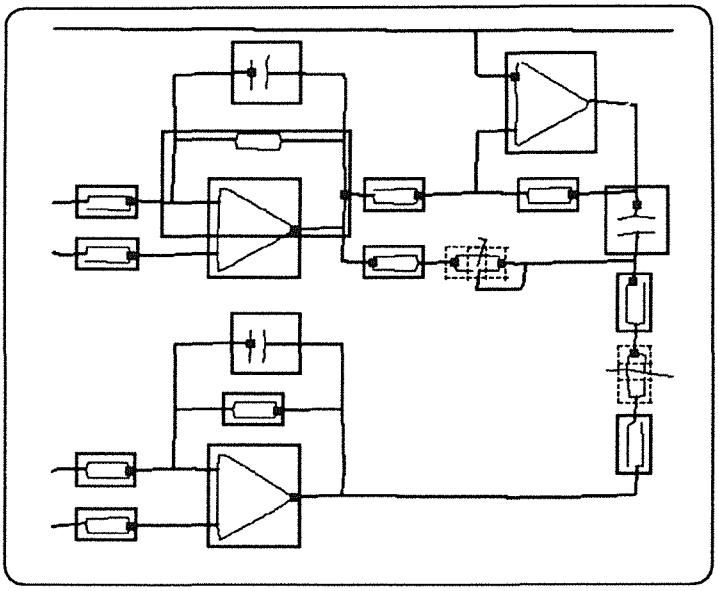
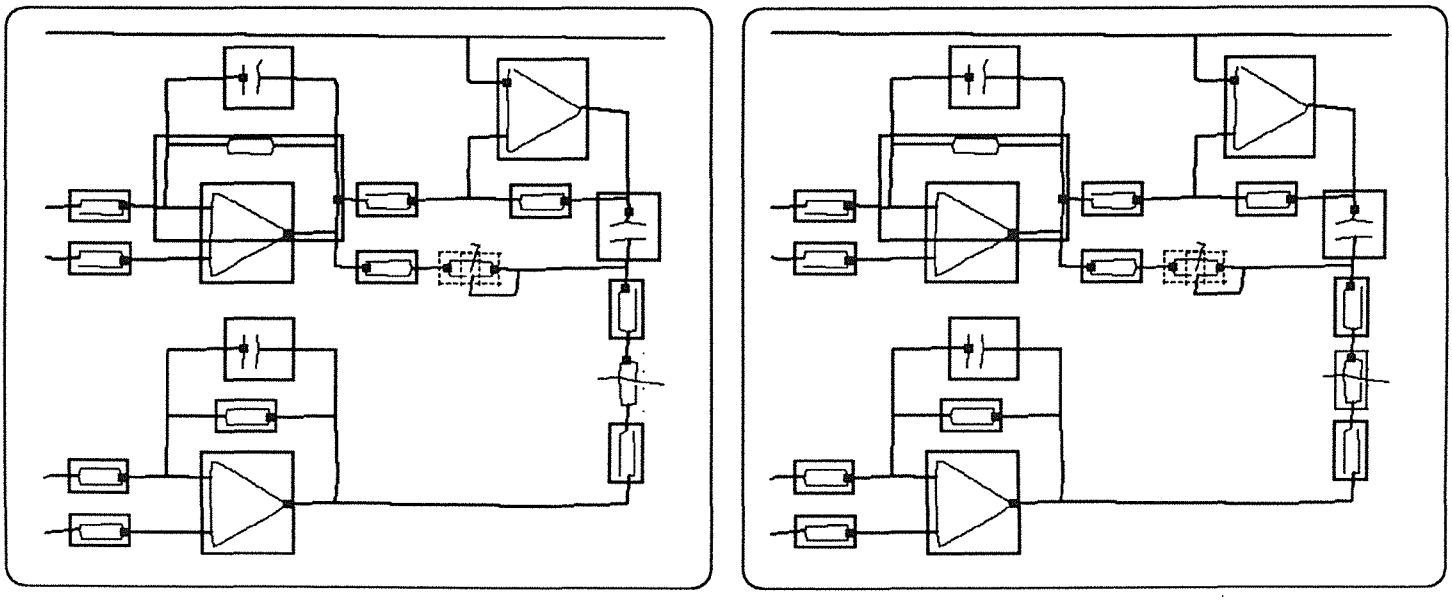
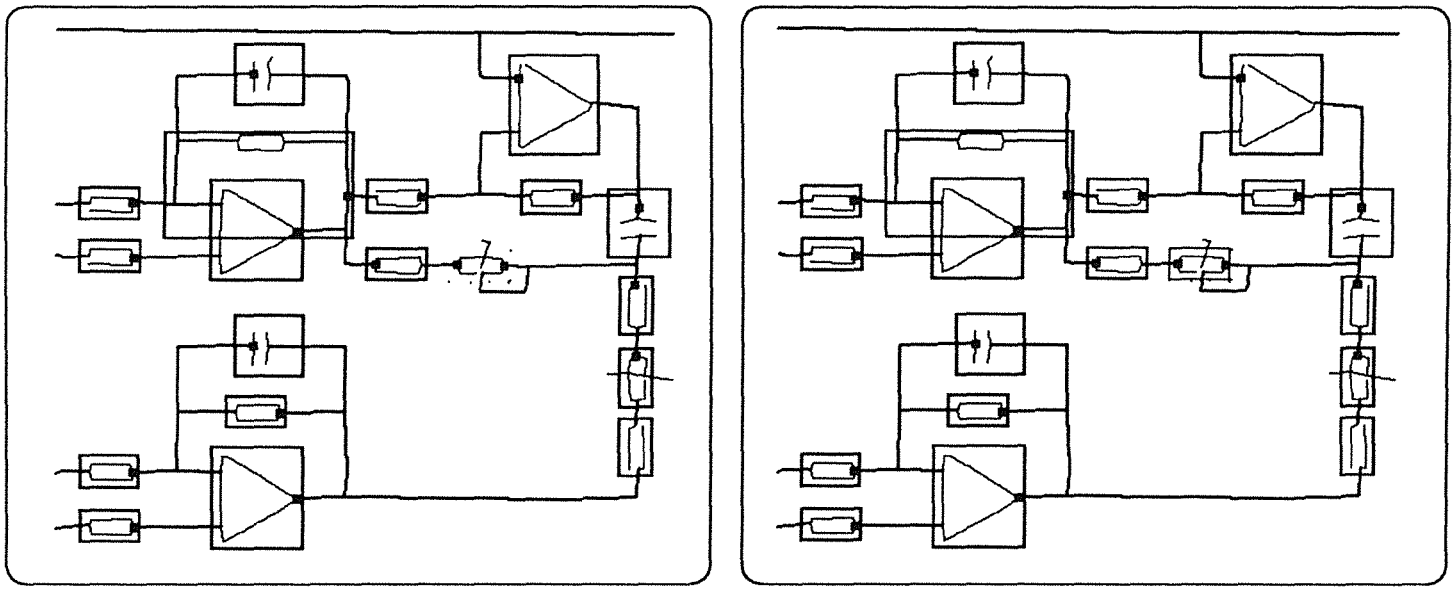


Figure 5.26. *Toutes les étapes d'analyses sont épuisées.*



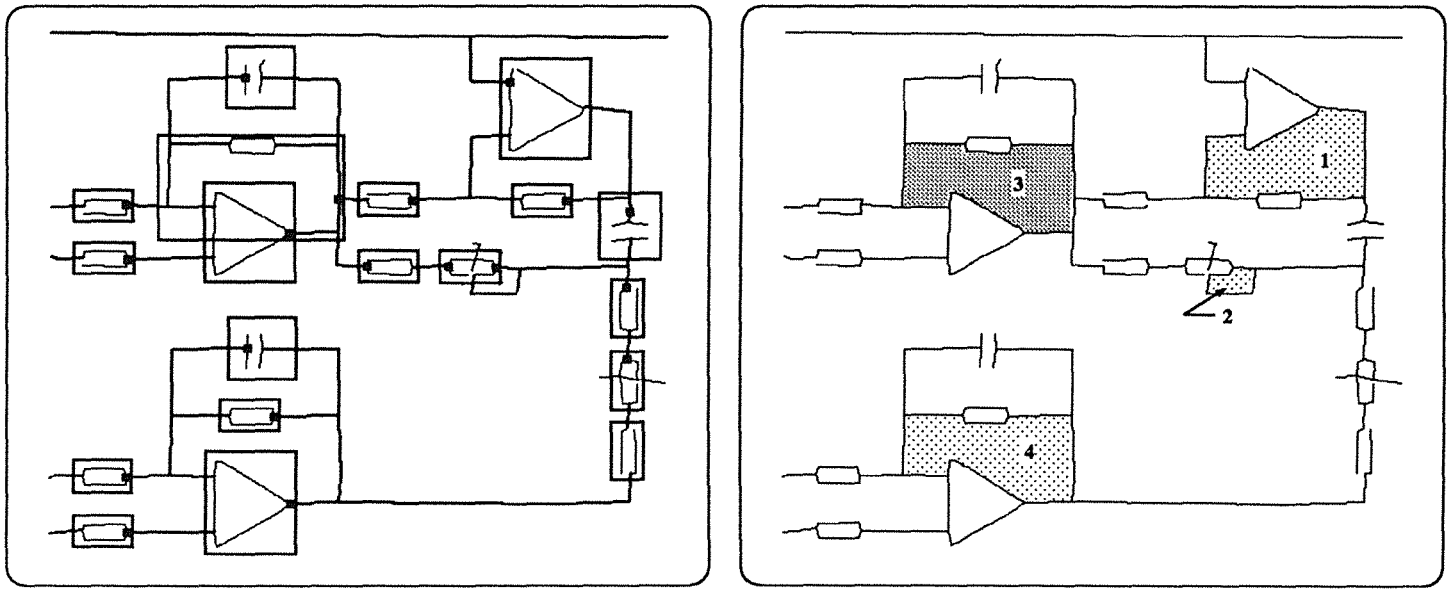
(a) (b)

Figure 5.27. *Etude des boucles problématiques et activation de l'agrandissement.*



(a) (b)

Figure 5.28. *Agrandissement du deuxième symbole problématique.*

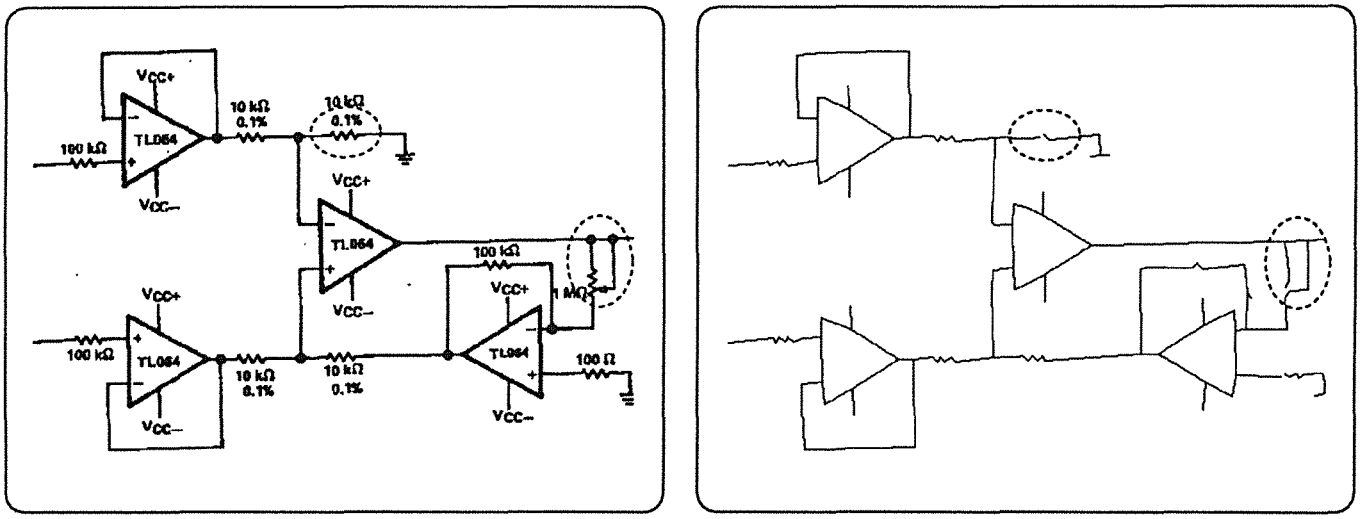


(a) (b)

Figure 5.29. *(a) Résultat final de l'analyse contextuelle du schéma, (b) Réduction du nombre de fausses boucles utilisées avant leurs détections.*

5.6.2 Cas d'activation par l'application de la règle (S_4) de construction d'hypothèses de présence de symboles

La Figure 5.30 montre le schéma à analyser ainsi que l'ensemble de segments obtenus après application des processus du bas niveau. Remarquons que le résultat du bas niveau est très affecté par le bruit. En effet, toutes les résistances sont très mal vectorisées. Par exemple les parties encerclées de l'image originale (Figure 5.30.a) sont complètement déformées dans l'image vectorisée (Figure 5.30.b).

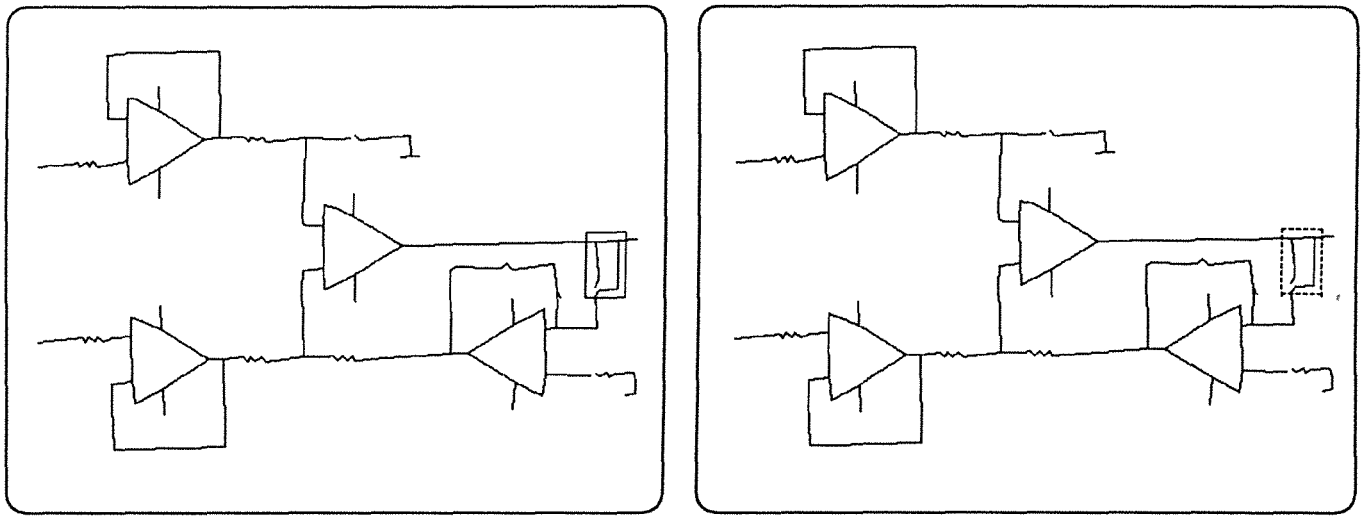


(a) *(b)*

Figure 5.30. *(a) Image originale, (b) Ensemble de segments obtenus après application du processus du bas niveau sur l'image (a).*

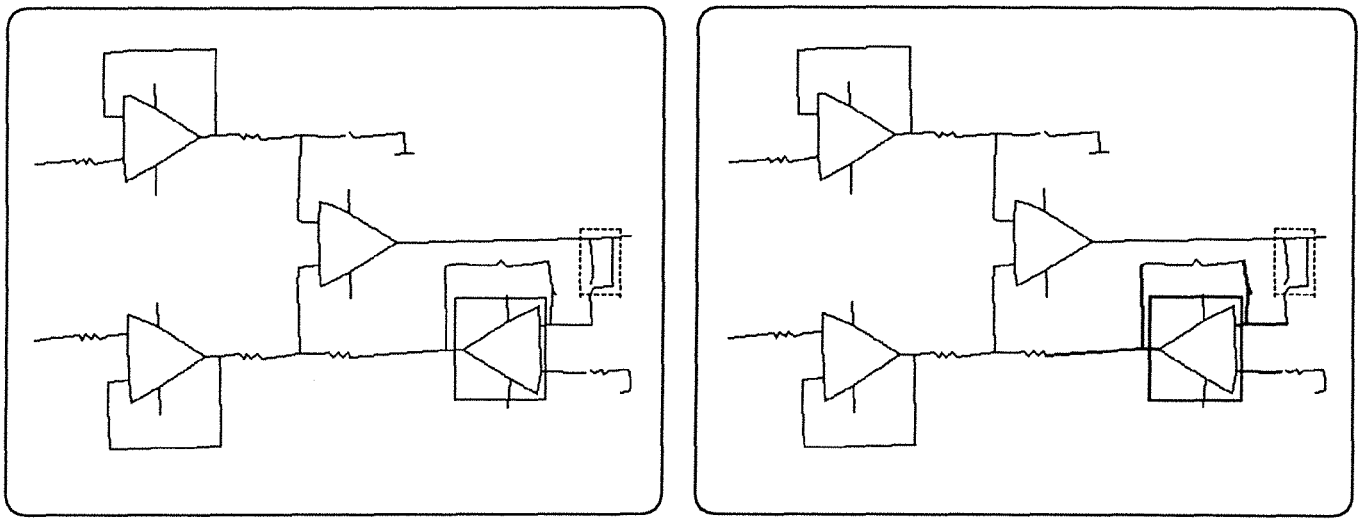
Après étude de la répartition des longueurs des lignes de connexion ne bordant pas des boucles, le module d'initialisation de l'ensemble des lignes de connexion échoue. Ceci s'explique par le fait que dans ce schéma, les lignes de connexion possèdent des longueurs comparables à celles des symboles et par conséquent aucun partitionnement n'a été fait. L'activation du système est alors effectuée par la sélection d'une zone candidate en utilisant uniquement l'information boucle, ce qui revient à l'application de la règle (S_4). Ceci est illustré par la Figure 5.31.a où apparaît la zone sélectionnée sans l'utilisation d'aucune information liée aux points extrémités des lignes de connexion. Remarquons que la boucle utilisée est une fausse boucle et du fait que le symbole, normalement contenu dans cette zone, est détérioré aucun symbole n'a été reconnu. Par conséquent cette boucle se transforme en une boucle problématique (Figure 5.31.b). A cette étape de l'analyse le seul traitement possible est toujours l'application de la règle (S_4) puisque l'ensemble des lignes de connexion n'est toujours pas initialisé. Cette deuxième application permet de sélectionner une nouvelle zone (Figure 5.32.a). La reconnaissance de cette zone a permis

l'initialisation de l'ensemble des lignes de connexion (Figure 5.32.b).



(a) (b)

Figure 5.31. *(a) Activation du système par la règle (S_4) de sélection de zone candidate, (b) Détection d'une boucle problématique.*

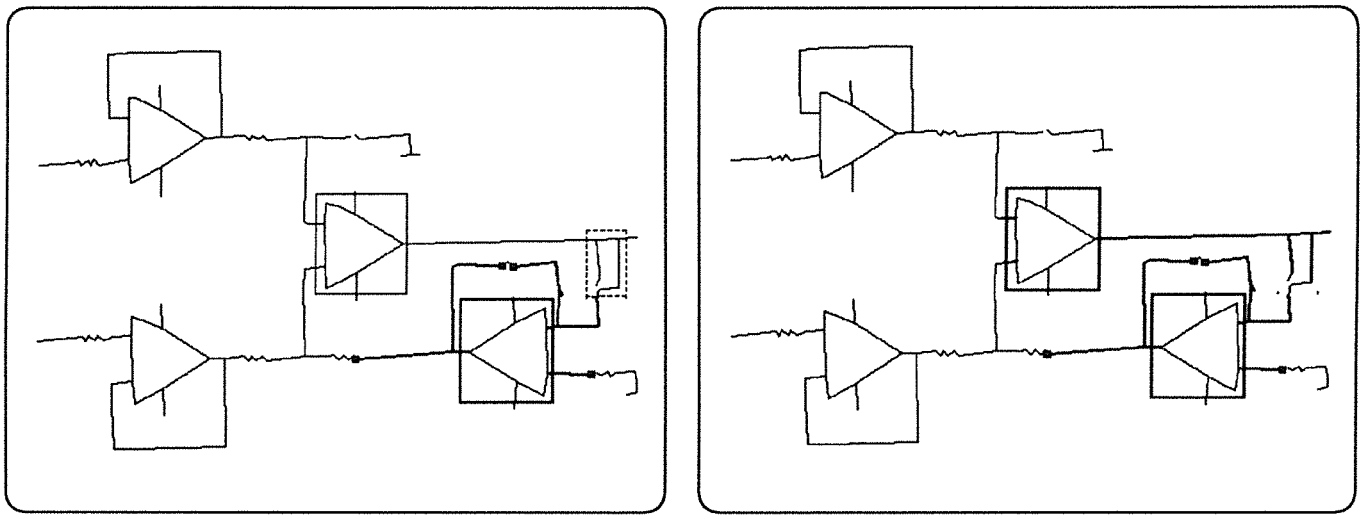


(a) (b)

Figure 5.32. *Initialisation de l'ensemble des lignes de connexion indirectement par l'application de la règle (S_4).*

Nous tenons à signaler que les boucles problématiques peuvent être ultérieurement détectées comme étant fausses. En effet, après reconnaissance de la zone sélectionnée de la Figure 5.33.a, la boucle broblématique est détectée comme étant une fausse boucle puisque l'une des nouvelles lignes de connexion la borde (Figure 5.33.b,

le rectangle en pointillé est supprimé). Nous faisons remarquer que l'agrandissement de cette boucle problématique dès l'échec de la reconnaissance n'aurait servi à rien. Cela confirme notre choix qui consiste à n'effectuer l'opération agrandissement que dans le cas où toutes les autres étapes d'analyse ont été épuisées. Par conséquent aucun agrandissement de fausses boucles ne sera fait.

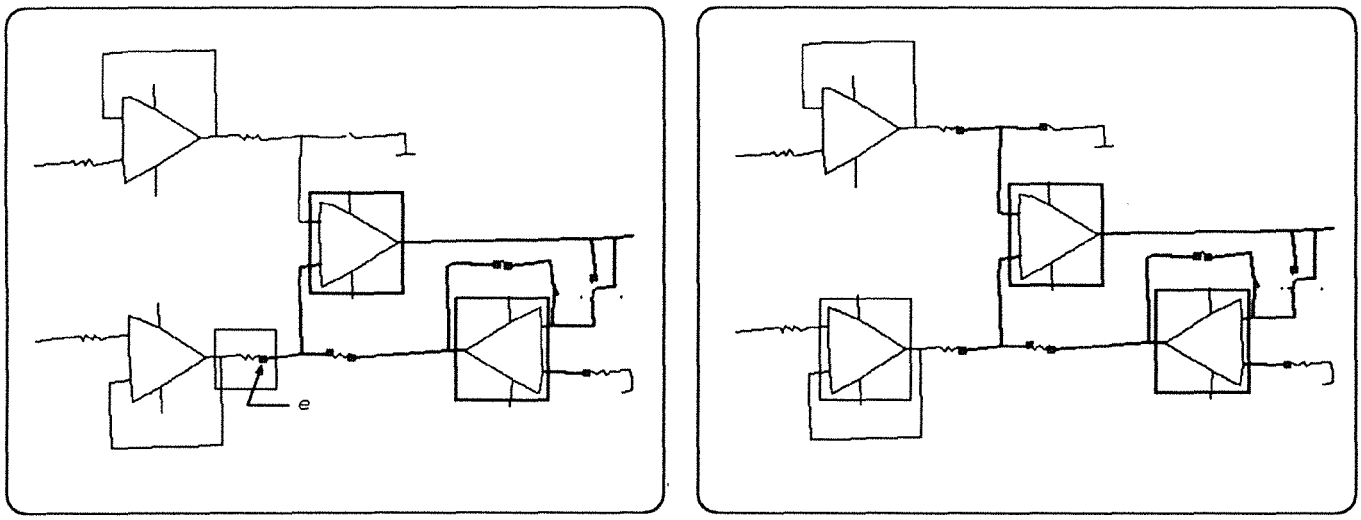


(a) *(b)*

Figure 5.33. *Détection de boucles problématiques comme étant fausses.*

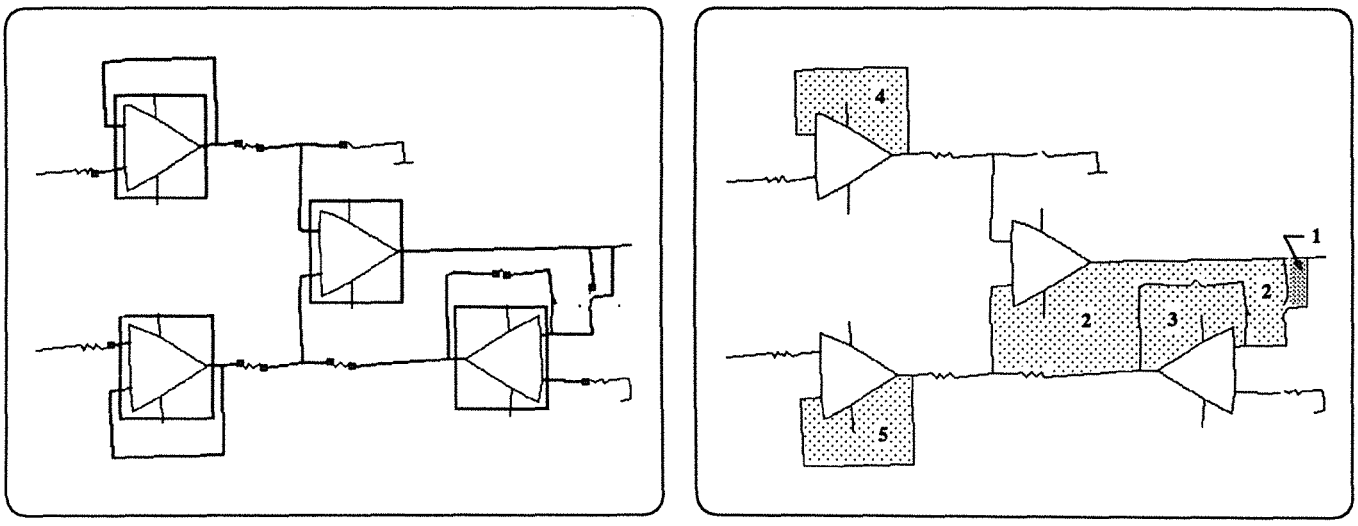
Dans le résultat expérimental présenté précédemment (§ 5.6.1), nous avons montré un cas d'échec de la reconnaissance sur des zones sélectionnées en utilisant les règles (S_1) ou (S_4). Mais dans l'exécution qui suit, la Figure 5.34 montre un cas d'échec de la reconnaissance sur une zone sélectionnée par la régle (S_2) où le point extrémité utilisé est considéré comme problématique. En effet, la zone à reconnaître de la Figure 5.34.a n'apparait plus dans la Figure 5.34.b du fait qu'elle n'est pas reconnue. Cet éhec est dû au taux élevé de bruit affectant les résistances. Le point extrémité e qui a permis sa sélection est alors un point problématique.

La Figure 5.35.a montre le résultat final. Tous les symboles contenant des boucles reconnus, ainsi que toutes les lignes de connexion ont été détectées. Par contre toutes les résistances ne sont pas reconnues. Les points problématiques, obtenus lors de l'analyse de ce schéma, sont tous ceux représentés par des petits rectangles noirs et n'appartenant pas à des zones reconnues. Ce fait montre l'influence de la qualité du processus du bas niveau sur les résultat de l'analyse. Dans ces cas, une étude plus raffinée de la zone problématique est nécessaire afin de pouvoir reconnaître ces symboles. La Figure 5.29.b montre qu'une seule fausse boucle du schéma a été utilisée (boucle numéro **1**) avant d'être détectée comme étant une fausse boucle.



(a) (b)

Figure 5.34. *Cas d'échec de la reconnaissance en engendrant un point problématique.*



(a) (b)

Figure 5.35. *(a) Résultat final de l'analyse contextuelle du schéma, (b) Réduction du nombre des fausses boucles utilisées avant leurs détections.*

5.6.3 Autres résultats expérimentaux

La Figure 5.36.c montre le résultat final de l'analyse du schéma de la Figure 5.36.a. Nous tenons à signaler que les lignes de connexion sont de longueur comparable par rapport à celle des symboles. Bien que les lignes de connexion ne sont pas très longues, nous avons réussi à les détecter. Ceci comfirme le fait que la restriction effectuée sur la longueur des lignes de connexion dans la plupart des travaux n'est pas toujours vérifiée. La Figure 5.36.d montre les trois fausses boucles utilisées (boucles numéro **2,3,5**) avant d'être détectées comme étant des fausses boucles.

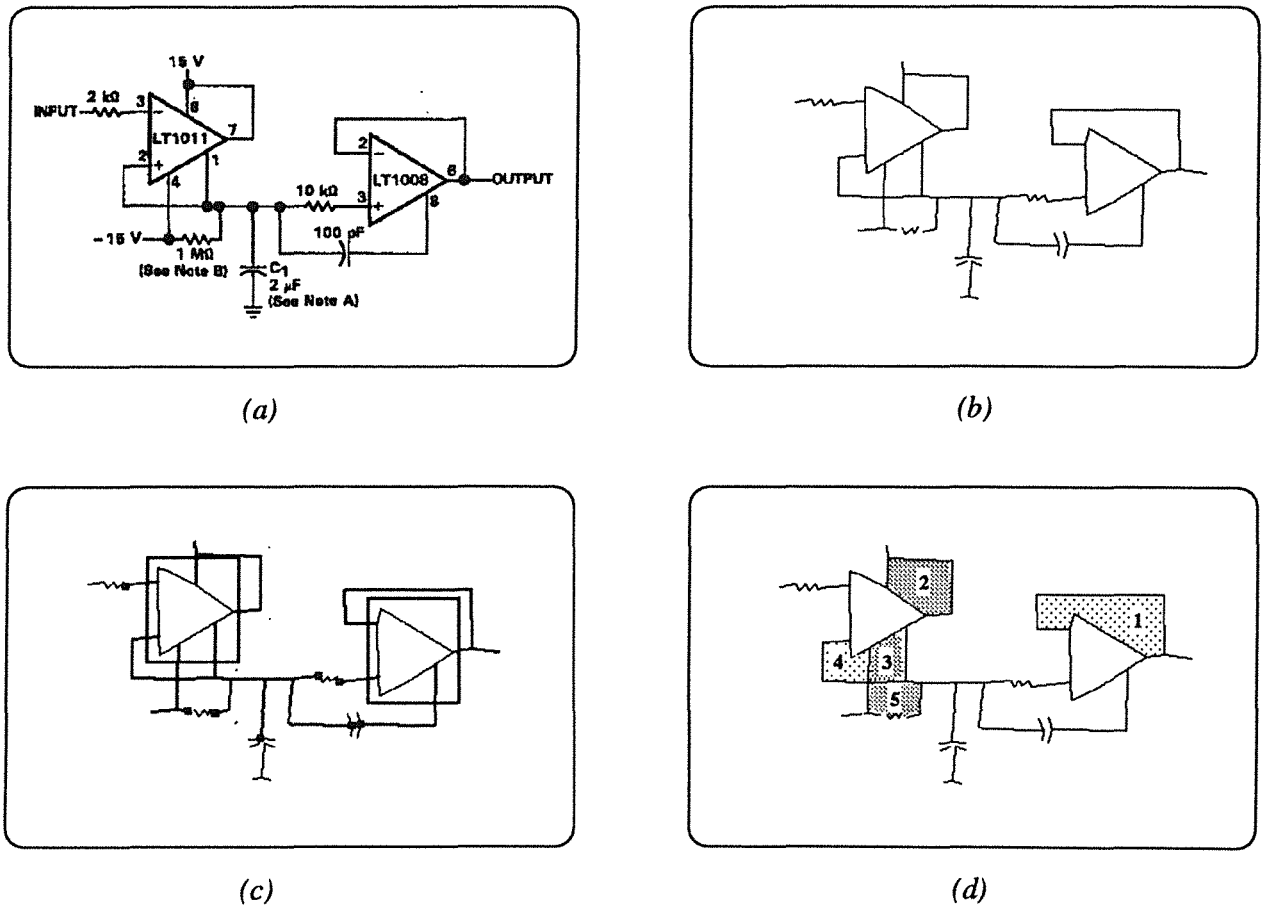


Figure 5.36. *(a) Image originale, (b) Ensemble de segments obtenus après application du processus du bas niveau sur (a), (c) Résultat final de l'analyse contextuelle du schéma de (a), (d) Fausses boucles utilisées lors de l'analyse.*

La Figure 5.37.d montre le résultat final de l'analyse du schéma de la Figure 5.37.a. Nous tenons à signaler que ce schéma fournit un contre exemple pour les restrictions concernant les directions des lignes de connexion, faites dans la plupart des travaux. En effet, il existe deux lignes de connexion qui n'ont ni la direction verticale, ni horizontale mais malgré cela, nous avons réussi à les détecter. La Figure 5.37.d montre

qu'une seule boucle parmi les neuf fausses boucles du schéma a été utilisée (boucle numéro **5**) avant d'être détectée comme étant une fausse boucle.

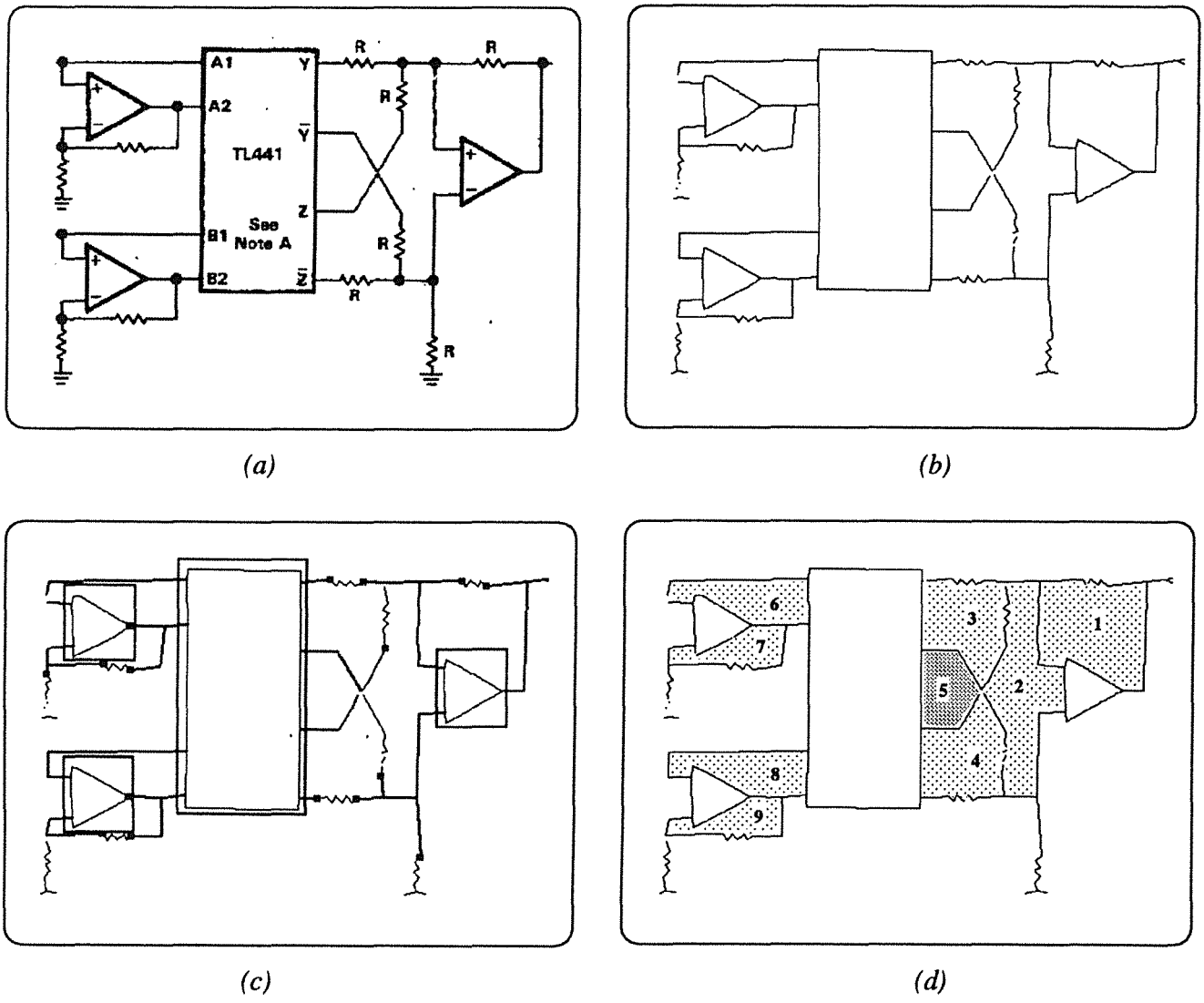


Figure 5.37. *(a) Image originale, (b) Ensemble de segments obtenus après application du processus du bas niveau sur (a), (c) Résultat de l'analyse contextuelle du schéma (a), (d) Fausses boucles utilisées lors de l'analyse.*

5.7 En résumé

Dans ce chapitre, nous avons présenté le processus d'interprétation de notre système. Dans un premier temps, nous avons décrit la construction d'hypothèses de présence de symboles. Ensuite, nous avons détaillé les différents modules permettant la détection des lignes de connexion. Enfin, nous avons décrit le fonctionnement du

système complet et validé les différentes étapes de notre approche par les résultats expérimentaux.

La principale originalité de notre approche réside dans l'utilisation des liens contextuels qui existent entre les trois sortes de composantes constituant un schéma. Cette utilisation des liens contextuels se traduit par une forte coopération entre les différents modules constituant le processus d'interprétation. Ceci a permis une détection des lignes de connexion sans aucune restriction sur leurs longueurs et sur leurs directions. Nous avons vu expérimentalement l'intêret d'une telle détection. Par ailleurs, nous avons réduit le nombre de fausses boucles traitées. En effet, la sélection des zones candidates n'est pas faite systématiquement sur toutes les boucles du schéma comme c'est le cas dans la plupart des travaux. En plus, l'utilisation du texte nous a permis de guider la sélection des symboles ne contenant pas de boucles. Dans les résultats expérimentaux, nous avons localisé tous ces symboles mais la majorité n'a pas été reconnue. Ce phénomène est dû à la qualité du résultat des processus de bas niveau, ce qui met en évidence l'importance de ces processus. Enfin, nous tenons à signaler que notre choix qui consiste à n'effectuer l'opération agrandissement que dans le cas où toutes les autres étapes d'analyse ont été épuisées, nous a permis d'éviter l'agrandissement de boucles problématiques détectées à une étape ultérieure comme étant fausse.

6
Conclusion et perspectives

Le travail présenté dans ce mémoire s'inscrit dans le domaine d'analyse de documents techniques et plus précisément présente un système d'interprétation automatique de schéma. La première partie de ce mémoire décrit le processus de vérification des hypothèses de présence de symboles. Ensuite, le problème de la détection des lignes de connexion est abordé, ainsi que celui de la construction des hypothèses de présence de symboles. Enfin, la description du fonctionnement du processus du haut niveau est donnée permettant ainsi de comprendre les interactions entre ces différents modules. Il convient alors de faire le bilan de notre travail tant sur le plan de l'étape d'interprétation du schéma complet que sur le plan de la reconnaissance de symboles isolés (section 6.1). Pour conclure ce travail de thèse, une étude des perspectives est effectuée dans la section 6.2.

6.1 Conclusion

Nous avons développé un système d'analyse automatique de schémas. Ce système est composé d'un bas niveau et d'un haut niveau. Les processus de bas niveau permettent l'extraction du *texte* et des *boucles caractéristiques* ainsi que la vectorisation de l'image. Ils utilisent la segmentation en composantes connexes du système REDRAW et la squelettisation, ainsi que l'approximation polygonale du logiciel VISILOG. D'après notre expérience, nous tenons à signaler l'importance des processus de bas niveau et par conséquent, l'importance de leur robustesse et de leur *adaptation aux documents techniques*, puisqu'ils influent automatiquement sur la qualité des résultats du processus de haut niveau, c'est à dire sur la qualité du système en entier.

Le haut niveau ou le processus d'interprétation proprement dit, effectue la détection *des lignes de connexion* d'une part et celle *des symboles* d'autre part. La détection des lignes de connexion est guidée par le processus de *reconnaissance*

de symboles. La détection de symboles est fondée sur le principe de *prédiction-vérification* d'hypothèses. La construction d'hypothèses de présence de symboles constitue l'étape de prédiction et délimite une zone candidate susceptible de contenir un symbole. Pour cela nous avons utilisé l'information d'existence de symboles aux extrémités des lignes de connexion déjà détectées et l'information boucles caractéristiques ou texte afin d'appuyer les hypothèses de présence de symboles aux points extrémités. Nous avons alors proposé quatre règles de sélection de zones candidates (S_1), (S_2), (S_3) et (S_4). Les trois premières sont fondées sur les points extrémités et traduisent les liens contextuels existant entre les lignes de connexion, les symboles et le texte. En revanche, la règle (S_4) utilise uniquement les boucles caractéristiques et permet l'extraction de symboles complètement isolés du schéma. Remarquons que dans le cas où l'ensemble des lignes de connexion n'a pas pu être initialisé, cette règle permet également l'activation du système. Nous avons instauré un ordre de priorité sur ces règles de sélection : $(S_1) > (S_2) > (S_3) > (S_4)$. La règle (S_4) est la moins prioritaire afin de favoriser l'utilisation des points extrémités.

Nous avons alors exploité au maximum les *liens contextuels* existants entre les trois composantes du schéma : le texte, les lignes de connexion et les symboles. En d'autres termes, la présence du texte implique une forte probabilité de présence de symboles à son voisinage, de même qu'à l'extrémité des lignes de connexion. Réciproquement, les lignes reliant un symbole déjà reconnu sont interprétées comme des lignes de connexion. Ceci constitue la principale originalité de ce travail qui réside dans l'utilisation des *liens contextuels.* Cette analyse contextuelle qui implique une forte coopération entre les trois modules constituant le processus d'interprétation permet de remédier aux différents points faibles dégagés des travaux existants dans ce domaine (§ 2.3). Parmi les avantages de cette coopération, citons :

- *l'efficacité de la détection des lignes de connexion :* cette détection est en partie guidée par le processus de reconnaissance. De plus aucune restriction n'a été faite ni sur la direction, ni sur la longueur de ces lignes ;

- *la réduction du nombres de fausses boucles traitées :* ceci provient du fait que la construction des hypothèses n'est pas faite à systématiquement sur toutes les boucles caractéristiques. L'utilisation des lignes de connexion détectées a permis d'éliminer les fausses boucles ayant comme contour des lignes de connexion ;

- *la localisation des symboles ne contenant pas de boucles :* elle est en partie réalisée à partir de l'information texte.

Nous avons également prévu de remettre en cause le résultat de l'étape de construction d'hypothèses de présence de symboles dans le cas d'un échec de la

reconnaissance sur la zone sélectionnée. Suivant la règle utilisée, le point extrémité ou la boucle caractéristique est considéré comme problématique. Un agrandissement est réalisé en étudiant le voisinage de la zone problématique afin de remédier au problème de sélection d'une sous partie du symbole. Nous avons choisi de réaliser l'agrandissement si et seulement si aucune des quatre règles n'est applicable et qu'aucune nouvelle ligne de connexion n'a pu être détectée. Et ce afin d'éviter d'agrandir des boucles problématiques qui s'avèrent fausses au cours de l'interprétation.

La *reconnaissance d'un symbole*, supposé contenu dans la zone candidate, permet la vérification de l'hypothèse de présence de symbole dans cette zone. Elle est fondée sur la mise en correspondance structurelle des graphes attribués et invariante par rapport à l'homothétie et à la rotation entre le symbole candidat et le modèle. Cette nécessité d'invariance provient du fait que d'une part la taille d'un symbole peut changer d'un schéma à l'autre et d'autre part deux occurrences d'un même symbole appartenant au même schéma peuvent avoir deux orientations différentes. L'originalité de cette approche de mise en correspondance réside dans l'utilisation d'un maximum d'informations *structurelles* (*topologiques* et *géométriques*). En effet, nous avons proposé trois contraintes pour réaliser la mise en correspondance de graphes attribués :

- une contrainte de *sélection de correspondants potentiels* qui détermine l'ensemble des sommets candidats homologues pour chaque sommet du modèle en utilisant l'information topologique et géométrique,
- une contrainte de *connexité* qui vérifie la conservation de la structure topologique du modèle dans le candidat,
- une contrainte de *disposition relative* qui vérifie la préservation de la structure géométrique du modèle dans le candidat.

Pour satisfaire ces contraintes, nous avons développé une *stratégie hiérarchique* permettant la vérification de la cohérence locale par une technique de relaxation discrète[1], avant de tester la cohérence globale de l'appariement obtenu. De plus les graphes attribués sont enrichissables par d'autres attributs symboliques ou numériques et ces trois contraintes sont définies de telle sorte qu'elles tiennent compte *du bruit affectant les symboles* (informations manquantes ou mal extraites).

Par ailleurs, les deux modules de détection de symboles, à savoir la construction d'hypothèses de présence de symboles et la reconnaissance de symboles candidats,

1. L'algorithme utilisé est AC4.

peuvent facilement évoluer vers la reconnaissance d'autres types de symboles comme ceux contenant des zones noires (cf § 6.2).

Mais, il subsiste le problème d'indexation de la base des modèles qui n'a pas été traité dans ce mémoire, bien qu'une première classification ait été faite en séparant les modèles contenant des boucles des autres Ceci ne réduit pas pour autant l'intérêt que nous portons à ce problème qui sera abordé dans les perspectives (cf § 6.2).

Au niveau implantation, notre travail a consisté à développer un système complet permettant l'analyse d'un schéma en reconnaissant les deux composantes graphiques qui le constituent. Ce travail a été réalisé en langage C et utilise l'interface Sunview. Il a donné lieu à deux logiciels ayant le même processus de bas niveau qui utilise la segmentation en composantes connexes du système REDRAW et la squelettisation, ainsi que l'approximation polygonale du logiciel VISILOG. Les deux logiciels obtenus sont :

- l'analyse contextuelle de schémas qui bien évidemment intègre le module de reconnaissance,
- la reconnaissance interactive de symboles qui possède sa propre interface et peut être utilisée indépendamment du système d'analyse du schéma complet.

6.2 Perspectives

D'après le bilan que nous venons de dresser, nous envisageons deux sortes de perspectives. Dans un premier temps, nous traitons le problème de l'évolution du système permettant l'analyse de schémas contenant des zones noires. Dans un deuxième temps, nous envisageons d'étudier le problème d'indexation de la base des modèles. Enfin, la reconnaissance de la composante texte des schémas est introduite.

A- Traitement des symboles contenant des zones noires

C'est une perspective à court terme. Le traitement des symboles contenant des zones noires nécessite une vectorisation des images par une technique permettant la détection des zones noires comme par exemple la technique d'appariement de contours, afin de ne pas déformer la structure des symboles. L'extension doit être prise en compte au niveau des deux modules de notre système constituant la détection des symboles :

- *la construction d'hypothèses :* nous proposons deux règles de sélection de zone candidate, définies par analogie aux règles (S_1) et (S_4) utilisant l'information

boucle caractéristique. La première règle (S'_1), fondée sur les points extrémités, renforce l'hypothèse de présence de symbole par l'existence d'une zone noire au voisinage. Par conséquent, si e est un point extrémité qui est connexe à une zone noire, la zone sélectionnée est définie comme étant le rectangle contenant le point e et la zone noire. La deuxième règle (S'_2) utilise uniquement l'information zone noire. Dans ce cas la zone sélectionnée est définie comme étant le rectangle contenant la zone noire. Bien évidemment les règles utilisant les points extrémités sont plus prioritaires que cette dernière règle. Remarquons que comme pour les boucles caractéristiques, une extraction des zones noires par le bas niveau est indispensable.

- *la reconnaissance de symboles :* du fait que les graphes attribués sont enrichissables par des attributs numériques ou symboliques, une zone noire est définie par sa position (gauche ou droite) par rapport aux segments qui la bordent. Par conséquent, une arête dans le graphe est représenté, en plus de son type (arc de cercle ou segment de droite), par un attribut symbolique donnant la position de la zone noire si elle en borde une. Au niveau de l'appariement, ces attributs seront traités dans la contrainte de connexité qui préserve les informations topologiques. En effet, nous exigeons que l'adjacence soit faite avec une arête qui aura les mêmes attributs que l'arête du modèle.

B- Indexation de la base des modèles

L'indexation de la base des modèles est une perspective à long terme qui nécessite une étude approfondie du problème d'indexation. L'indexation revient à attribuer à un symbole candidat un sous-ensemble de la base des modèles (dans le cas idéal, un seul modèle) afin d'appliquer l'étape de vérification qui est la mise en correspondance sur un nombre réduit de modèles. Nous avons déjà commencé par aborder ce problème en effectuant une première étude bibliographique dont voici la présentation :

- **les moments invariants :** la méthode des moments invariants constitue un moyen de caractériser les images par sept coefficients, appelés les moments invariants, composés de combinaisons de moments statistiques d'ordre deux et trois. Ces coefficients sont invariants par rapport à la translation, à l'homothétie, à la rotation, à la réflexion et à la combinaison de ces transformations.

  Chaque classe de modèles est définie par le vecteur des sept moments invariants. L'image est attribué à la classe qui minimise la distance entre les vecteurs représentant les modèles et le vecteur représentant l'objet. La représentation des images par les moments invariants est très sensible aux bruits,

et plus l'ordre est élevé plus les moments sont affectés par le bruit. D'après les tests effectués par [Halet 91], la méthode des moments invariants donne de bons résultats pour des images à texture complexe et par conséquent, pour des images à niveau de gris ; or ce n'est pas notre cas.

- **les tables de hachage :** elles constituent un moyen d'indexation très utilisé dans plusieurs domaines. Le principe de cette technique est bien illustré par les travaux de Sossa *et al.* [Sossa 92] qui utilisent des tables de hachage pour l'indexation d'objets 3D. Dans ce travail, les objets sont modélisés par leurs *vues caractéristiques 2D* qui sont décomposées en *figures fondamentales*, sous graphes de vues caractéristiques. Chaque figure fondamentale est décrite par un vecteur de coefficients $(c_0, \ldots, c_{n-1})$ de sa forme "$d_2$polynomiale" qui permet de caractériser le graphe. Les figures fondamentales et leurs intersections sont les entrées des tables de hachage. La table utilisée est à deux niveaux et seuls les coefficients c_0 et c_{n-1} sont utilisés dans ce travail à cause de la complexité du calcul de ces coefficients.

  C'est une méthode sensible aux erreurs car le bruit affecte le calcul des clefs de hachage et par conséquent, influe énormément sur le résultat de l'indexation dans le cas où les images sont bruitées.

- **les graphes de décision :** un graphe de décision représente les concepts qui caractérisent un ensemble de données qui forment une classification hiérarchique [Burns 87] [Paris 92].

  Dans les travaux de Paris [Paris 92] effectués au sein de notre équipe, l'index est un graphe qui exprime une classification *structurelle* hiérarchique des modèles d'objets. Les objets modélisés sont des aspects 3D et les entrées de l'index sont les relations qui existent entre les primitives : par exemple la relation de connexité entre faces. Le processus d'indexation est incrémental puisqu'il réorganise l'index pour chaque nouvel aspect, en utilisant le graphe d'indexation existant. Cette organisation de la classification structurelle hiérarchique est automatique. Mais la taille du graphe d'indexation peut être importante, ce qui influe sur le coût de la reconnaissance. C'est une méthode sensible aux erreurs car le bruit affecte la détermination des relations structurelles reliant les primitives.

Pour résoudre ce problème, il nous a paru intéressant d'étudier le comportement de l'indexation par des tables de hachage et par des graphes de décision sur nos modèles. Les tables de hachage conservent l'information zone de présence alors que les graphes de décision ne l'utilisent pas. Mais, les tables de hachage sont plus rigides que les les garphes de décision. Aussi, il nous parait intéressant de tester quelle méthode est la plus efficace et la plus robuste pour notre problème d'indexation.

C- La reconnaissance du texte

Dans ce mémoire, nous avons uniquement utiliser le texte pour guider la prédiction de présence de symboles ne contenant pas de boucles. Pour que le schéma soit entièrement analysé, il reste alors à reconnaître le texte qui est la troisième composante d'un schéma. De plus la reconnaissance de texte peut dans certains cas intervenir dans l'indexation de la base de modèles. En effet, nous avons remarqué que le premier caractère du texte avoisinant un symbole est le premier caractère du nom du symbole. Par exemple dans la Figure 6.1.a, le texte inscrit au voisinage de la résistance commence par le caractère 'R'. Dans le cas où seule la valeur associée au symbole est indiquée comme dans la Figure 6.1.b, nous pouvons envisager la reconnaissance de l'unité qui permettrait de sélectionner le modèle.

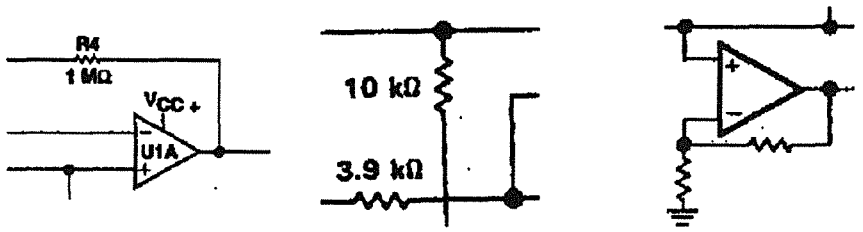


Figure 6.1. *Utilisation de la reconnaissance du texte pour identifier le symbole.*

Par conséquent, à part le fait que la présence du texte implique celle d'un symbole, la reconnaissance du texte permet de réduire l'ensemble des modèles sur lequelle nous appliquons la mise en correspondance. Bien évidement, en aucun cas notre système ne doit devenir complètement dépendant de la reconnaissance du texte puisqu'il faut toujours prévoir le cas où il n'y a pas de texte au voisinage du symbole (Figure 6.1.c). Ainsi la reconnaissance du texte lorsque celui-ci existe, doit être utilisée comme une information complémentaire et non fondamentale dans notre système.

Annexe A
Algorithme de consistance d'arcs AC4

L'algorithme AC4 [Mohr 86] permet de vérifier la consistance locale d'un graphe étiqueté. Nous avons utilisé cet algorithme pour vérifier la consistance locale des correspondants des sommets du graphe modèle par rapport à nos contraintes de connexité et de disposition relative définies sur des couples de sommets adjacents.

A.1 Données de l'algorithme

Un problème de propagation de contraintes est défini par un graphe étiqueté et une relation binaire à propager.

- $N = \{n_i, n_j, \ldots\}$: ensemble de nœuds avec $|N| = n$.
- $A = \{(n_i, n_j),\ n_i, n_j \in N\}$: ensemble des arcs avec $|A| = s$.
- $E = \{e_a, e_b, \ldots\}$: ensemble des étiquettes avec $|E| = e$.
- $E_i = \{e_b /\ e_b \in E$ *et* e_b *est une étiquette de* $n_i\}$: ensemble des étiquettes admissibles au nœud n_i.
- $\Re$ est une relation unaire : *si* $\Re(n_i, e_a)$ *alors* (n_i, e_a) *est admissible.*
- $\Re'$ est une relation binaire : *si* $\Re'(n_i, e_a, n_j, e_b)$ *alors* $(n_i, e_a)-(n_j, e_b)$ *est admissible.*

A.2 Algorithme AC4

L'idée générale consiste à ne considérer que les étiquettes affectées par l'élimination d'une autre étiquette. Pour chaque couple (n_i, e_a), une liste de couples (n_j, e_b)

qui vérifie que l'étiquette e_a du nœud n_i est consistante avec l'etiquette e_b du nœud n_j est construite. Cette liste est notée par $S_{(n_i,e_a)}$:

$$S_{(n_i,e_a)} = \{(n_j, e_b) \ / \ \mathcal{R}'(n_i, e_a, n_j, e_b) = vrai\}$$

On mémorise aussi, pour chaque nœud n_j, le nombre d'étiquettes e_b de ce nœud qui sont compatible avec l'étiquette e_a du nœud n_i :

$$C[n_i, e_a, n_j] = |\{e_b \ / \ \mathcal{R}'(n_i, e_a, n_j, e_b) = vrai\}|$$

En plus de ces deux structures de données, on marque les étiquettes inconsistantes donc celles qui ne vérifient pas la relation $\Re'$:

$$PILE = \{(n_i, e_a) \ / \ e_a \ est \ non \ valide \ pour \ n_i\}$$

Cet algorithme s'exécute en deux étapes. Dans la première étape, On construit la structure de données. La deuxième étape, consiste à l'élagage des étiquettes inconsistantes.

A.2.1 Construction de la structure de données

Cette étape consiste à l'initialisation des trois ensembles définis auparavant. La construction de cette structure de données est réalisée par la fonction suivante :

Fonction *Initialisation*
$PILE \longleftarrow \emptyset$
Pour $(e_a, n_i) \in E_i \times N$ Faire $S_{e_a,n_i} \longleftarrow \emptyset$ Fin pour
Pour $(n_i, n_j) \in A$ Faire
Pour $e_a \in E_i$ Faire
$total \longleftarrow 0$
Pour $e_b \in E_j$ Faire
Si $\Re(n_i, e_a, n_j, e_b)$
Alors $total \longleftarrow total + 1$
$S_{e_b,n_j} \longleftarrow S_{e_b,n_j} \cup \{(e_a, n_i)\}$
Fin pour
Si $total = 0$
Alors $E_i \longleftarrow E_i - \{e_a\}$
$PILE \longleftarrow PILE \cup \{(e_a, n_i)\}$
Sinon $C[e_a, n_i, n_j] \longleftarrow total$
Fin pour
Fin pour
Fin Fonction

A.2.2 Elagage des étiquettes inconsistantes

Cette étape effectue la propagation de la relation $\mathcal{R}'$ en élagant les étiquettes inconsistantes. Cet élagage opère sur la structure de donnée préparée par l'étape d'initialisation et est effectué par la fonction suivante :

Fonction *Elagage*
  Tant que $PILE \neq \emptyset$ Faire
    choisir (n_k, e_c) *dans* $PILE$
    $PILE \longleftarrow PILE - \{(n_k, e_c)\}$
    Pour $(n_j, e_b) \in S_{e_c,n_k}$ Faire
      $C[e_b, n_j, n_k] \longrightarrow C[e_b, n_j, n_k] - 1$
      Si $C[e_b, n_j, n_k] = 0$ *et* $e_b \in E_j$
      Alors $E_j \longleftarrow E_j - \{e_b\}$
        $PILE \longleftarrow PILE \cup \{(n_j, e_b)\}$
      Fsi
    Fin pour
  Fin tantque
Fin Fonction

Annexe B
Processus d'interprétation en terme de règles d'inférence

A chaque moment, le système est complètement défini par l'état du schéma décrit par :

- les symboles reconnus
- les lignes de connexion détectées
- les points extrémités
- les points de branchement
- les boucles caractéristiques pertinentes et le texte.
- les fausses boucles.
- les cas problématiques : points et boucles problématiques.

Pour cela nous définissons notre système comme un ensemble de transitions du schéma d'un état à un nouveau état ; cet ensemble de transitions sera décrit en termes de *règle d'inférence*. Dans un premier temps, nous présentons la structure de données en définissant le descripteur d'un état du système et deuxièment les règles d'inférence réalisant la transition d'un descripteur au descripteur suivant et par conséquent, décrivant le fonctionnement du processus d'interprétation de notre système d'analyse de schéma.

B.1 Structure de données

Un descripteur du système est défini par les douze ensembles suivants :

$$(S, L_t, L_n, B, E, B_p, E_{pb}, B_{pb}, B_f, E^f_{pb}, B^f_{pb}, Z) \quad \text{où :}$$

- S: ensemble des symboles reconnus.
- L_t: ensemble des lignes de connexion détectées et telles que leurs points de branchement sont traités.
- L_n: ensemble des lignes de connexion detectées telles que leur point de branchement sont non traités.
- B: ensemble des points de branchement.
- E: ensemble des points extrémités des lignes de connexion.
- B_p: ensemble des boucles caractéristiques pertinentes. B_p est initialisé par toutes les boucles caractéristiques de l'image.
- B_f: ensemble des fausses boucles.
- E_{pb}: ensemble des points extrémités problématiques non encore étudiés.
- B_{pb}: ensemble des boucles problématiques non encore étudiées.
- E_{pb}^f: ensemble des points extrémités problématiques même après agrandissement.
- B_{pb}^f: ensemble des boucles problématiques même après agrandissement.
- Z: cet ensemble contient la zone candidate ainsi que le numéro de la règle qui a permis sa sélection.

B.2 Les règles d'inférence

Une règle d'inférence est représentée par un quotient de deux descripteurs: le descripteur courant (au numérateur) et le nouveau atteint en appliquant cette règle (au dénominateur). Pour des raisons de clarté, nous avons choisi de ne pas encombrer la représentation. Ainsi, pour chaque règle nous présentons dans le descripteur uniquement les ensembles qui sont utiles à la règle et donc modifiés par la règle ou qui définissent une condition à vérifier pour l'application de cette règle. Les ensembles qui n'apparaissent pas dans le descripteur ne sont pas affectés par cette règle. Notons que ces règles d'inférence sont appliquées dans l'ordre et répétées (sauf la règle d'initialisation) jusqu'à la fin de l'analyse.

- **Initialisation**

$$\frac{(L_n = \emptyset, E = \emptyset)}{(L_n = \{Part(schema)\}, E = \{Pt\_extremite(Part(schema))\})} \quad \text{(B.1)}$$

La fonction $Part(schema)$ étudie le partitionnement des lignes du schéma et retourne, dans le cas du possible, les lignes de connexions détectées sinon L reste vide. La fonction $Pt\_extremite(L)$ fournit les points extrémités de la lignes de connexion L (§ 5.4.1).

- **Détection de fausses boucles**

$$\frac{(L_n = L'_n \cup \{l\}, B_p = B'_p \cup \{fb\}, B_f)}{(B'_p, B_f = B_f \cup \{fb\})} \; si\; l \in Contour(fb) \quad \text{(B.2)}$$

$$\frac{(L_n = L'_n \cup \{l\}, B_{pb} = B'_{pb} \cup \{fb\}, B_f)}{(B'_{pb}, B_f = B_f \cup \{fb\})} \; si\; l \in Contour(fb) \quad \text{(B.3)}$$

$Contour(fb)$ retourne l'ensemble des lignes de connexion qui bordent la boucle fb.

- **Suivi des points de branchement**

$$\frac{(L_t, L_n = L'_n \cup \{l\}, B = \emptyset)}{(L_t = L_t \cup \{l\}, L'_n, B = \{Pt\_branchPt(l)\})} \quad \text{(B.4)}$$

$$\frac{(B = B' \cup \{b\}, E = E' \cup \{b\})}{(B', E')} \quad \text{(B.5)}$$

$$\frac{(L_n, B = B' \cup \{b\}, E)}{(L_n = L_n \cup \{l\}, B', E = E \cup \{Pt\_extremite(l)\})} \; si\; b \notin E. \quad \text{(B.6)}$$

La fonction $Pt\_branchPt(l)$ fournit les points des branchement de l (§ 5.4.3).

- **Sélection de la zone candidate par (S_1) ou (S_2) ou (S_3)**

$$\frac{(L_n = \emptyset, B = \emptyset, E = E' \cup \{e\}, Z = \emptyset, B_p)}{(L_n, B, E', Z = \{(S_1(e, b_p), 1)\}, B_p = B_p - \{b_p\})} \quad \text{(B.7)}$$

$$\frac{(L_n = \emptyset, B = \emptyset, E = E' \cup \{e\}, Z = \emptyset)}{(L_n, B, E', Z = \{(S_2(e), 2)\})} \; si\; S_1\; échoue \quad \text{(B.8)}$$

$$\frac{(L_n = \emptyset, B = \emptyset, E = E' \cup \{e_1, e_2\}, Z = \emptyset)}{(L_n, B, E', Z = \{(S_3(e_1, e_2), 3)\})} \; si\; S_1\; et\; S_2\; échoue \quad \text{(B.9)}$$

La fonction $S_1(e, b_p)$ (respectivement $S_2(e)$, $(S_3(e_1, e_2))$) fournit la zone candidate sélectionnée par la règle (S_1) (respectivement (S_2) et (S_3)) présentée dans § 5.3.1.

- **Suivi des points extrémités**

$$\frac{(L_n = \emptyset, B = \emptyset, E = E' \cup \{e\})}{(L_n = \{Suivi(e)\}, B, E = E' \cup \{Pt\_extremite(Suivi(e))\})} \; si\ S_1\ ,\ S_2\ et\ S_3\ échouent \tag{B.10}$$

La fonction $Suivi(e)$ fournit les lignes de connexion détectées par le suivi du point extrémité e (§ 5.4.4).

- **Reconnaissance**

$$\frac{(S, L_n = \emptyset, B = \emptyset, E, Z = \{(z, num)\})}{(S = S \cup \{Rec(z)\}, L_n = \{Ligne(z)\}, B, E = E \cup \{Pt\_extremite(Ligne(z))\}, Z = \emptyset)} \tag{B.11}$$

$$\frac{(L_n = \emptyset, B = \emptyset, Z = \{(z, num \in \{1,4\})\}, B_{pb})}{(L_n, B, Z = \emptyset, B_{pb} = B_{pb} \cup \{b_p\})} \; si\ Rec\ échoue \tag{B.12}$$

$$\frac{(L_n = \emptyset, B = \emptyset, Z = \{(z, num \in \{2,3\})\}, E_{pb})}{(L_n, B, Z = \emptyset, E_{pb} = E_{pb} \cup \{e\})} \; si\ Rec\ échoue \tag{B.13}$$

$Rec(z)$ retourne le symbole reconnu dans la zone candidate z et $Ligne(z)$ retourne les lignes de connexion détectées en utilisant le symbole reconnu (§ 5.4.2).

- **Sélection de zone candidate par S_4**

$$\frac{(L_n = \emptyset, B = \emptyset, E = \emptyset, Z = \emptyset, B_p = B_p' \cup \{b_p\})}{(L_n, B, E, Z = \{S_4(b_p)\}, B_p')} \tag{B.14}$$

$S_4(b_p)$ fournit une zone candidate sélectionnée par la règle (S_4) (§ 5.3.1).

- **Agrandissement**

$$\frac{(L_n = \emptyset, B = \emptyset, E = \emptyset, B_p = \emptyset, B_{pb} = B'_{pb} \cup \{b\}, B^f_{pb}, Z = \emptyset)}{(L_n, B, E, B_p, B'_{pb}, B^f_{pb}, Z = \{Agrand_1(b)\})} \tag{B.15}$$

$$\frac{(L_n = \emptyset, B = \emptyset, E = \emptyset, B_p = \emptyset, B_{pb} = B'_{pb} \cup \{b\}, B^f_{pb}, Z = \emptyset)}{(L_n, B, E, B_p, B'_{pb}, B^f_{pb} = B^f_{pb} \cup \{b\}, Z\})} \; si\ Agrand_1\ échoue \tag{B.16}$$

$Agrand_1(b)$ retourne une nouvelle zone candidate sélectionnée en agrandissant la boucle problématique b.

$$\frac{(L_n = \emptyset, B = \emptyset, E = \emptyset, B_p = \emptyset, E_{pb} = E'_{pb} \cup \{e\}, E^f_{pb}, Z = \emptyset)}{(L_n, B, E, B_p, E'_{pb}, E^f_{pb}, Z = \{Agrand_2(e)\})} \tag{B.17}$$

$$\frac{(L_n = \emptyset, B = \emptyset, E = \emptyset, B_p = \emptyset, E_{pb} = E'_{pb} \cup \{e\}, E^f_{pb}, Z = \emptyset)}{(L_n, B, E, B_p, E'_{pb}, E^f_{pb} = E^f_{pb} \cup \{e\}, Z\})} \; si\ Agrand_2\ échoue \tag{B.18}$$

$Agrand_2(e)$ retourne une nouvelle zone candidate sélectionnée en étudiant le voisinage du point problématique e.

- **Fin d'interprétation**

$$\frac{(E = \emptyset, B_p = \emptyset, E_{pb} = \emptyset, B_{pb} = \emptyset)}{\textbf{Arrêt}} \tag{B.19}$$

Bibliographie

[Anigbogu 92] J.C. Anigbogu. Reconnaissance de textes imprimés multifontes à l'aide des modèles stochastiques et métriques. Thèse de doctorat, Université de Nancy 1, Vandœuvre-lès-Nancy, Septembre 1992.

[Antoine 90] D. Antoine, S. Collin et K. Tombre. Analysis of Technical Documents: The REDRAW System. *Pre-proceedings of IAPR Workshop on Syntactic and Structural Pattern Recognition, Murray Hill, NJ (USA)*, pages 1–20, June 1990.

[Antoine 91] D. Antoine. Interprétation des plans cadastraux français à partir d'un modèle. Thèse de doctorat, Institut National Polytechnique de Lorraine, Vandœuvre-lès-Nancy, 1991.

[Aubert 91] M. Aubert, A. Chehikian et L. Delaporte. Mixed Image/Text Office Documents Processing. *Proceedings of First International Conference on Document Analysis, Saint-Malo, France*, volume 1, pages 192–200, 1991.

[Ayache 86] N. Ayache et O.D. Faugeras. HYPER: a New Approach for the Recognition and Positioning of 2D Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):44–54, 1986.

[Ayache 87] N. Ayache et B. Faverjon. Efficient Registration of Stereo Images by Matching Graph Descriptions of Edge Segments. *International Journal of Computer Vision*, 8(1):107–131, 1987.

[Baird 91] H.S. Baird et R. Fossey. A 100-Font Classifier. *Proceedings of First International Conference on Document Analysis, Saint-Malo, France*, volume 1, pages 332–340, 1991.

[Ballard 82] D.H. Ballard et C.M. Brown. *Computer Vision.* Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[Baptista 88] G. Baptista et K.M. Kulkarni. A High Accuracy Algorithm for Recognition of Handwritten Numerals. *Pattern Recognition*, 21(4):287–291, 1988.

[Belaïd 92] A. Belaïd et Y. Belaïd. *Reconnaissance des formes — Méthodes et applications.* InterEditions, Paris, 1992.

[Bhanu 84] B. Bhanu et O.D. Faugeras. Shape Matching of Two-Dimensional Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2):137–156, 1984.

[Bley 84] H. Bley. Segmentation and Preprocessing of Electrical Schematics Using Picture Graphs. *Computer Vision, Graphics and Image Processing*, 28:271–288, 1984.

[Bolles 82] R.C. Bolles et R.A. Cain. Recognizing and Locating Partially Visible Objects: The Local-Feature-Focus Method. *The International Journal of Robotics Research*, 1(3), 1982.

[Bråten 85] K. Bråten, E. Holbæk Hanssen et T. Taxt. A General Software System for Supervised Statistical Classification of Symbols. *Proceedings of International Seminar on Symbol Recognition, Oslo (Norway)*, pages 103–117, 1985.

[Bunke 82] H. Bunke. Attributed Programmed Graph and Application to Diagram Interpretation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(4):574–582, 1982.

[Bunke 90a] H. Bunke. Hybrid Pattern Recognition Methods. H. Bunke et A. Sanfeliu, éditeurs, *Syntactic and Structural Pattern Recognition: Theory and Applications*, chapitre 11, pages 308–347. World Scientific, 1990.

[Bunke 90b] H. Bunke. String Matching for Structural Pattern Recognition. H. Bunke et A. Sanfeliu, éditeurs, *Syntactic and Structural Pattern Recognition: Theory and Applications*, chapitre 5, pages 119–144. World Scientific, 1990.

[Burns 87] J. B. Burns et L. J. Kitchen. An Approach to Recognition in 2D Images of 3D Objects from Large Model Bases. Rapport no. COINS 87-85, University of Massachusetts at Amherst., 1987.

[Collin 92] S. Collin. Interprétation de la cotation des dessins techniques par analyse syntaxique. Thèse de doctorat, Institut National Polytechnique de Lorraine, Vandœuvre-lès-Nancy, Janvier 1992.

[Cugini 84] U. Cugini, G. Ferri, P. Mussio et M. Protti. Pattern-directed Restoration and Vectorization of Digitized Engineering Drawings. *Computers and Graphics*, 8(4):337–350, 1984.

[Duda 72] R.O. Duda et P.E. Hart. Use of the Hough Transform to Detect Lines and Curves in Pictures. *Communication of ACM*, 15:11–15, 1972.

[Duda 73] R.O. Duda et P.E. Hart. *Pattern Classification and Scene Analysis.* Wiley-InterScience, 1973.

[Duerr 79] B. Duerr, W. Haettich, H. Tropf et G. Winkler. A Combination of Statistical and Syntactical Pattern Recognition Applied to Classification of Unconstrained Handwritten Numerals. *Pattern Recognition*, 12:189–199, 1979.

[Fahmy 92] H. Fahmy et D. Blostein. Graph Grammar Processing of Uncertain Data. *IAPR International Workshop on Structural and Syntactic Pattern Recognition, Bern (Switzerland)*, 1992.

[Fahn 88] C.S. Fahn, J.F. Wang et J.Y. Lee. A Topology-Based Component Extractor for Understanding Electronic Circuit Diagrams. *Computer Vision, Graphics and Image Processing*, 44:119–138, 1988.

[Faugeras 81] O.D. Faugeras et M. Berthod. Improving Consistency and Reducing Ambiguity in Stochastic Labeling : an Optimization Approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4:412–423, 1981.

[Fekete 81] G. Fekete, J.O. Eklundh et A. Rosenfeld. Relaxation: Evaluation and Applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(4):459–469, 1981.

[Fletcher 88] L.A. Fletcher et R. Kasturi. A Robust Algorithm for Text String Separation from Mixed Text/Graphics Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):910–918, 1988.

[Fu 74] K.S. Fu. *Syntactic Methods in Pattern Recognition.* Academic Press, New York, 1974.

[Fukushima 88] K. Fukushima. A Neural Netwok for Visual Pattern Recognition. *IEEE COMPUTER Magazine*, 21(3):65–75, March 1988.

[Futatsumata 90] T. Futatsumata, G. Shichino, J. Shibayama et A. Maeda. Development of an Automatic Recognition System for Plant Diagrams. *Proceedings of IAPR Workshop on Machine Vision Applications, Tokyo*, pages 207–210, 1990.

[Girod 92] J.P. Girod et J. Brémont. Utilisation de réseaux neuromémiques, en visionique, pour la reconnaissance de caractères. *Actes du deuxième Colloque National sur l'Ecrit et le Document, Nancy, France*, pages 302–309, 1992.

[Granger 85] C. Granger. Reconnaissance d'objets par mise en correspondance en vision par ordinateur . Thèse de doctorat, Université de Nice, 1985.

[Groen 84] F.C.A. Groen et R.J. van Munster. Topology Based Analysis of Schematic Diagrams. *Proceedings of 7th International Conference on Pattern Recognition, Montreal*, volume 2, pages 1310–1312, 1984.

[Groen 85] F.C.A. Groen, A. C. Sanderson et J. F. Schlag. Symbol Recognition in Electrical Diagrams Using Probabilistic Graph Matching. *Pattern Recognition Letters*, 3:343–350, 1985.

[Habacha 91a] A.H. Habacha. Structural Recognition of Disturbed Symbols Using Discrete Relaxation. *Proceedings of First International Conference on Document Analysis, Saint-Malo, France*, volume 1, pages 170–178, 1991.

[Habacha 91b] A.H. Habacha et K. Tombre. Structural Symbol Recognition. *Proceedings of 7th Scandinavian Conference on Image Analysis, Aalborg (Denmark)*, volume 1, pages 486–493, August 1991.

[Habacha 92] A.H. Habacha. Une approche invariante par similitude pour la reconnaissance des symboles bruités dans un document technique. *Actes du deuxième Colloque National sur l'Ecrit et le Document, Nancy, France*, pages 264–272, 1992.

[Halet 91] A. Halet, H.T. Huynh et G. Lampropoulos. La classification des images affectées par le bruit avec la méthode des moments invariants homoge'nes. *Actes du 13ème Colloque sur le Traitement du Signal et des Images* (GRETSI), *Juan-lès-Pins*, Septembre 1991.

[Haralick 80] R.M. Haralick et G.L. Elliot. Increasing Tree Search Efficiency for Constraint Satisfaction Problems. *Artificial Intelligence*, 14:263–313, 1980.

[Harris 82] J.F. Harris, J. Kittler, B. Llwellyn et G. Preston. A Modular System for Interpreting Binary Pixel Representations of Line-Structured Data. L.F. Pau J. Kittler, K.S. Fu, éditeur, *Pattern Recognition Theory and Applications*, pages 311–351. D. Reidel Publishing Company, 1982.

[Haton 89] J.P. Haton et M.C. Haton. *L'intelligence artificielle.* Numéro 2444 in Que sais-je? Presses Universitaires de France, 1989.

[Henderson 90] T.C. Henderson. *Discrete Relaxation Techniques.* Oxford University Press, 1990.

[Horaud 89] R. Horaud et T. Skordas. Stereo Correspondance Through Feature Grouping and Maximal Cliques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(11):1168–1180, November 1989.

[Illingworth 88] J. Illingworth et J. Kittler. A Survey of the Hough Transform. *Computer Vision, Graphics and Image Processing*, 44:87–116, 1988.

[Kittler 87] J. Kittler. Statistical Pattern Recognition: The State of the art. V. Cantoni, V. Di Gesù et S. Levialdi, éditeurs, *Image Analysis and Processing*, volume 2, pages 57–66. Plenum Press, 1987.

[Kohonen 89] T. Kohonen. *Self-Organization And Associative Memory.* Springer Verlag, 3 édition, 1989.

[Kuner 85] P. Kuner. Efficient Techniques to Solve the Subgraph Isomorphism Problem for Pattern Recognition in Line Images. *Proceedings of 4th Scandinavian Conference on Image Analysis, Trondheim (Norway)*, pages 333–340, 1985.

[Kuner 86] P. Kuner et B. Ueberreiter. Knowledge Based Pattern Recognition in Disturbed Line Images Using Graph Theory Optimization and Predicate Calculus. *Proceedings of 8th International Conference on Pattern Recognition, Paris*, pages 240–243, 1986.

[Landy 85] M.S. Landy et Y. Cohen. Vectorgraph Coding: Efficient Coding of Line Drawings. *Computer Vision, Graphics and Image Processing*, 30:331–344, 1985.

[Laurière 86] J.L. Laurière. *Intelligence artificielle.* Eyrolles, 1986.

[Lee 90] S.W. Lee, J.H. Kim et F.C.A. Groen. Translation-, Rotation- and Scale-Invariant Recognition of Hand-Drawn Electrical Circuit Symbols with Attributed Graph Matching. *Pre-proceedings of IAPR Workshop on Syntactic and Structural Pattern Recognition, Murray Hill, NJ (USA)*, pages 273–292, 1990.

[Li 92] S.Z. Li, J. Kittler et M. Petrou. Matching and Recognition of Road Networks from Aerial Images. G. Sandini, éditeur, *Proceedings of Second European Conference on Computer Vision, Santa Margherita Ligure (Italy)*, pages 857–861, 1992. Lecture Notes in Computer Science, Vol. 588.

[Lin 85] X. Lin, S. Shimotsuji, M. Minoh et T. Sakai. Efficient Diagram Understanding with Characteristic Pattern Detection. *Computer Vision, Graphics and Image Processing*, 30:84–106, 1985.

[Lux 85] A. Lux. *Algorithmique et contrôle en vision par ordinateur.* Thèse d'état, Université Scientifique et Médicale de Grenoble, Institut National Polytechnique de Grenoble, 1985.

[Mackworth 85] A.K. Mackworth et E.C. Freuder. The Complexity of Some Polynomial Network Consistency Algorithms for the Constraint Satisfaction Problems. *Artificial Intelligence*, 25:65–74, 1985.

[Mohr 86] R. Mohr et T.C. Henderson. Arc and Path Consistency Revisited. *Artificial Intelligence*, 28:225–233, 1986.

[Mohr 87] R. Mohr. A Correct Path Consistency Algorithm and an Optimal Generalized Arc Consistency Algorithm. Rapport CRIN no. 87-R-030, Centre de Recherche en Informatique de Nancy, Vandœuvre-lès-Nancy, 1987.

[Mohr 88a] R. Mohr. Sur l'appariement modèle-perception. *Actes du 2ème Atelier Scientifique, traitement d'images: du pixel à l'interprétation*, pages XXIX–1–XXIX–7, Aussois,France,1988.

[Mohr 88b] R. Mohr et G. Masini. Good Old Discrete Relaxation. *Proceedings of the 8th European Conference on Artificial Intelligence, Munich*, pages 651–656, 1988.

[Mohr 90] R. Mohr, T. Pavlidis et A. Sanfeliu, éditeurs. *Structural Pattern Analysis*, volume 19, série *Series in Computer Science.* World Scientific, 1990.

[Muller-Belaïd 84] Y. Muller-Belaïd. *Détection de surfaces par transformée de Hough en vision tridimentionnelle.* Thèse de $3^{ème}$ Cycle en informatique, Université de Nancy 1, 1984.

[Niemann 90] H. Niemann. *Pattern Analysis and Understanding.* Springer Verlag, 2 édition, 1990.

[Okazaki 88] A. Okazaki, T. Kondo, K. Mori, S. Tsunekawa et E. Kawamoto. An Automatic Circuit Diagram Reader with Loop-Structure-Based Symbol Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(3):331–341, 1988.

[Ouladj 89] H. Ouladj, G. Lorette, E. Petit, J. Lemoine et M. Gaudaire. From Primitives to Letters. A Structural Method to Automatic Cursive handwritting Recognition. *Proceedings of 6th Scandinavian Conference on Image Analysis, Oulu (Finland)*, pages 593–598, 1989.

[Paris 88] S. Paris. Un algorithme généralisé pondéré de relaxation discrète. Rapport de dea, université de nancy i, Centre de Recherche en Informatique de Nancy, Vandœuvre-lès-Nancy, 1988.

[Paris 92] S. Paris. *Reconnaissance par indexation en vision par ordinateur.* Thèse de doctorat, Institut National Polytechnique de Lorraine, Vandœuvre-lès-Nancy, Novembre 1992.

[Quan 89] L. Quan. Contribution de la vision monoculaire à la perception tridimensionnelle. Thèse de doctorat, Institut National Polytechnique de Lorraine, octobre 1989.

[Sabah 88] G. Sabah. *L'intelligence artificielle et le langage — représentation des connaissances*, volume 1. Hermès, 1988.

[Sanfeliu 86] A. Sanfeliu et H. Bunke. General Remarks on Syntactic Pattern Recognition. *Pattern Recognition*, 19(4):249–254, 1986.

[Sanfeliu 90] A. Sanfeliu. Matching Trees Structures. H. Bunke et A. Sanfeliu, éditeurs, *Syntactic and Structural Pattern Recognition: Theory and Applications*, chapitre 6, pages 145–178. World Scientific, 1990.

[Shapiro 80] L.G. Shapiro et R.M. Haralick. Algorithms for Inexact Matching. *Proceedings of 5th International Joint Conference on Pattern Recognition, Miami*, pages 202–207, 1980.

[Shapiro 81] L.G. Shapiro et R. Haralick. Structural Description and Inexact Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(5):504–519, 1981.

[Sossa 92] H. Sossa et R. Horaud. Model Indexing: The Graph-hashing Approach. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Urbana Champaign, (USA)*, pages 811–814, 1992.

[Tai 90] J.W. Tai et Y.J. Liu. Chineese Character Recognition. H. Bunke et A. Sanfeliu, éditeurs, *Syntactic and Structural Pattern Recognition: Theory and Applications*, chapitre 15, pages 416–451. World Scientific, 1990.

[Tombre 87] K. Tombre. La saisie automatisée de documents composites : reconnaissance, codage et interprétation des parties graphiques. Thèse de doctorat, Institut National Polytechnique de Lorraine, Vandœuvre-lès-Nancy, 1987.

[Tsai 80] W.H. Tsai et K.S. Fu. Attributed Grammar: A Tool for Combining Syntactic and Statistical Approaches to Pattern Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(12):873–885, 1980.

[Tsai 90] W.H. Tsai. Combining Statistical and Structural Methods. H. Bunke et A. Sanfeliu, éditeurs, *Syntactic and Structural Pattern Recognition: Theory and Applications*, chapitre 12, pages 349–366. World Scientific, 1990.

[Ullmann 76] J.R. Ullmann. An Algorithm for Subgraph Isomorphism. *Journal of the ACM*, 23(1):31–42, 1976.

[Wall 84] K. Wall et P. Danielsson. A Fast Sequential Method for Polygonal Approximation of Digitized Curves. *Computer Vision, Graphics and Image Processing*, 28:220–227, 1984.

[You 80] K.C. You et K.S. Fu. Distorted Shape Recognition Using Attributed Grammars and Error-Correcting Techniques. *Computer Vision, Graphics and Image Processing*, 13:1–16, 1980.

[Ziou 91] D. Ziou. La détection de contours dans des images à niveaux de gris : mise en œuvre et sélection de détecteurs. Thèse de doctorat, Institut National Polytechnique de Lorraine, Vandœuvre-lès-Nancy, 1991.

Liste des Figures

INSTITUT NATIONAL
POLYTECHNIQUE
DE LORRAINE

AUTORISATION DE SOUTENANCE DE THESE DU DOCTORAT DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE

o0o

VU LES RAPPORTS ETABLIS PAR :
Monsieur NIEMANN Heinrich, Professeur Université Nurnberg (RFA)
Monsieur LORETTE Guy, Professeur, IRISIA Rennes,
Professeur HATON J.Paul, Professeur, CRIN/INRIA Vandoeuvre.

NANCY BRABOIS
2, AVENUE DE LA
FORET-DE-HAYE
BOITE POSTALE 3
F - 5 4 5 0 1
VANDŒUVRE CEDEX

Le Président de l'Institut National Polytechnique de Lorraine, autorise :

Mademoiselle HABACHA-HAMADA Anja

à soutenir devant l'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE, une thèse intitulée :
"Reconnaissance de symboles techniques et analyse contextuelle de schémas"

en vue de l'obtention du titre de :

DOCTEUR DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE

Spécialité : **"INFORMATIQUE"**

INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE NANCY

Fait à Vandoeuvre le, 14 Juin 1993
Le Président de l'I.N.P.L.,
M. LUCIUS

Pour le Président
Le Vice-Président
J. Ch. CHEVRIER

TEL. 33/83.59.59.59
FAX. 33/83.59.59.55

Résumé

Dans le contexte plus général de l'analyse de documents techniques, notre objectif consiste, d'une part, à extraire toutes les composantes (texte, symboles et lignes de connexion) constituant un schéma et, d'autre part, à reconnaître les deux composantes graphiques. Pour cela, nous proposons une approche qui exploite au maximum les liens contextuels existant entre les trois sortes de composantes. Parmi les avantages de l'utilisation de ces liens contextuels, il y a l'efficacité de la détection des lignes de connexion et la réduction du nombre de fausses boucles traitées. En effet, la détection des lignes de connexion est totalement guidée par le résultat du processus de reconnaissance de symboles. Notons qu'aucune restriction n'est faite, ni sur la direction des lignes de connexion ni sur leur longueur. Le module de détection de symboles est fondé sur le principe de prédiction-vérification d'hypothèses. La construction d'hypothèses de présence de symbole utilise en plus de l'information points extrémités des lignes de connexion, l'information boucle ou texte pour prédire la présence de symboles. La vérification des hypothèses est réalisée par l'étape de reconnaissance de symboles. Pour cette étape, nous proposons une méthode originale de mise en correspondance de graphes attribués, à base d'étiquetage structurel et géométrique et de relaxation discrète pour assurer la consistance locale, avant vérification de la consistance globale. Cette approche est invariante par rapport à l'homothétie et la rotation qui existent entre le symbole et le modèle.

Liste de mots clés: Documets techniques, interprétation, lien contextuel, reconnaissance de symboles, mise en correspondance structurelle, détection des lignes de connexion, consistance locale, consistance globale.