Chapitre III :

Evaluation des organigrammes des apprenants en algorithmique

I. Introduction

Après avoir vu les limites des environnements d'apprentissage de l'algorithmique existants et leurs limites, nous proposons un système d'évaluation de l'algorithmique qui peut, très facilement, être adapté à l'évaluation de n'importe quel savoir-faire. Ce système est basé (inspiré de) sur la décomposition. Il garantit, en plus, l'évaluation de n'importe quelle solution de l'apprenant de manière directe ou différée.

Ce système repose sur la comparaison et l'appariement. Inspiré des concepts et des techniques d'appariement de modèles, il se concentre principalement sur les aspects structurels des solutions à apparier. A la fin, un prototype de ce système est présenté. Ce prototype va servir à la validation de notre approche.

II. Apprentissage par la pratique

La littérature met en lumière plusieurs stratégies d'apprentissage dont les plus analysées sont celles qui ont trait à l'apprentissage par la recherche (*learning by searching*), à l'apprentissage par la pratique (*learning by doing*), à l'apprentissage par l'utilisation des technologies de pointe (*learning by using*), à l'apprentissage par l'interaction (*learning by interacting*), à l'apprentissage par les externalités industrielles (*learningfromindustryspillovers*) et à l'apprentissage par les externalitésrégionales (*learningregion*) (Cohen 1990).

Dans le contexte de l'apprentissage de l'algorithmique et de la programmation, une matière qui requiert beaucoup de compétences, on trouve l'apprentissage par la pratique comme la forme la plus adéquate pour apprendre ce genre de compétence. Aristote a dit: « Il faut apprendre en faisant la chose, car si vous pensez que vous savez, vous n'avez pas la certitude jusqu'à ce que vous ayez essayé. »

Cet apprentissage par la pratique est le meilleur moyen d'apprendre la programmation puisque les apprenants ont besoin d'une meilleure compréhension de ce que l'algorithme signifie réellement. De plus, ils obtiennent une compréhension plus profonde de leurs algorithmes proposés surtout quand ils proposent une solution erronée et qu'ils reçoivent un feedback formatif (Labat, 2002).

Dans cette thèse, nous avons adopté l'approche *apprentissage par la pratique* et une stratégie qui vise à réutiliser des solutions communes et en essayant de trouver la solution la

plus similaire qui correspond à la solution proposée par l'apprenant à l'aide de mesure de similarité entre organigrammes. Dans la section suivante, nous expliquons la méthode d'évaluation proposée ainsi que l'outil proposé.

Qu'est-ce que l'organigramme et Pourquoi ?

Un organigramme est une représentation graphique normalisée, utilisée pour analyser ou décoder un problème. Les organigrammes sont utilisés pour :

- la Communication : Organigrammes sont une meilleure façon de communiquer la logique d'un système à tous les intéressés ou impliqués.
- Une analyse efficace : Avec l'aide de diagramme, un problème peut être analysé de manière plus efficace réduisant ainsi le coût et le gaspillage de temps.
- Une documentation adéquate : organigrammes du programme servent comme étant une bonne documentation du programme, qui est nécessaire à des fins diverses, ce qui rend les choses plus efficaces.
- Codage efficace : Les organigrammes agissent comme un guide ou un plan au cours de la phase d'analyse des systèmes et des programmes.
- Mise au point correcte : L'organigramme aide dans le processus de débogage.
- Maintenance du programme efficace : Le maintien du programme d'exploitation devient facile avec l'aide d'organigramme. Il aide le programmeur à mettre des efforts plus efficacement sur cette partie
- les symboles standards sont normalisés.
- Ils peuvent être compris par des personnes n'ayant aucune connaissance de la programmation.
- Ils peuvent servir à diviser le projet entier en sous-tâches. Ils serviront alors à évaluer les progrès accomplis.
- Ils décomposent les séquences d'opérations et aident ainsi à trouver les erreurs.

III. Modélisation d'une solution algorithmique

Pour simplifier des tâches complexes, il faut les décomposer en des tâches moins complexes et réitérer cette opération jusqu'à arriver à un niveau de décomposition comportant des opérations de base et/ou des opérations élémentaires. L'algorithme solution du problème sera alors une composition de ces dernières opérations (de base et élémentaires). Le nombre

d'étapes de décomposition dépend de la complexité du problème : plus ce dernier est complexe, plus le nombre d'étapes est important.

Cette méthode de raffinages successifs (dite également approche descendante) permet de passer progressivement, et avec un maximum de chances de réussite, de la description abstraite de la solution du problème (par une opération complexe) à l'algorithme qui permettra sa résolution. L'algorithme est au dernier niveau de raffinage lorsqu'il ne comporte que des opérations de base, des opérations élémentaires et des structures de contrôle.

"*A deep understanding of programming, in particular the notions of successive decomposition as a mode of analysis and debugging of trial solutions, results in signicant educational bene ts in many domains of discourse, including those unrelated to computers and information technology per se.*'' (Seymou Papert, in 'Mindstorms')

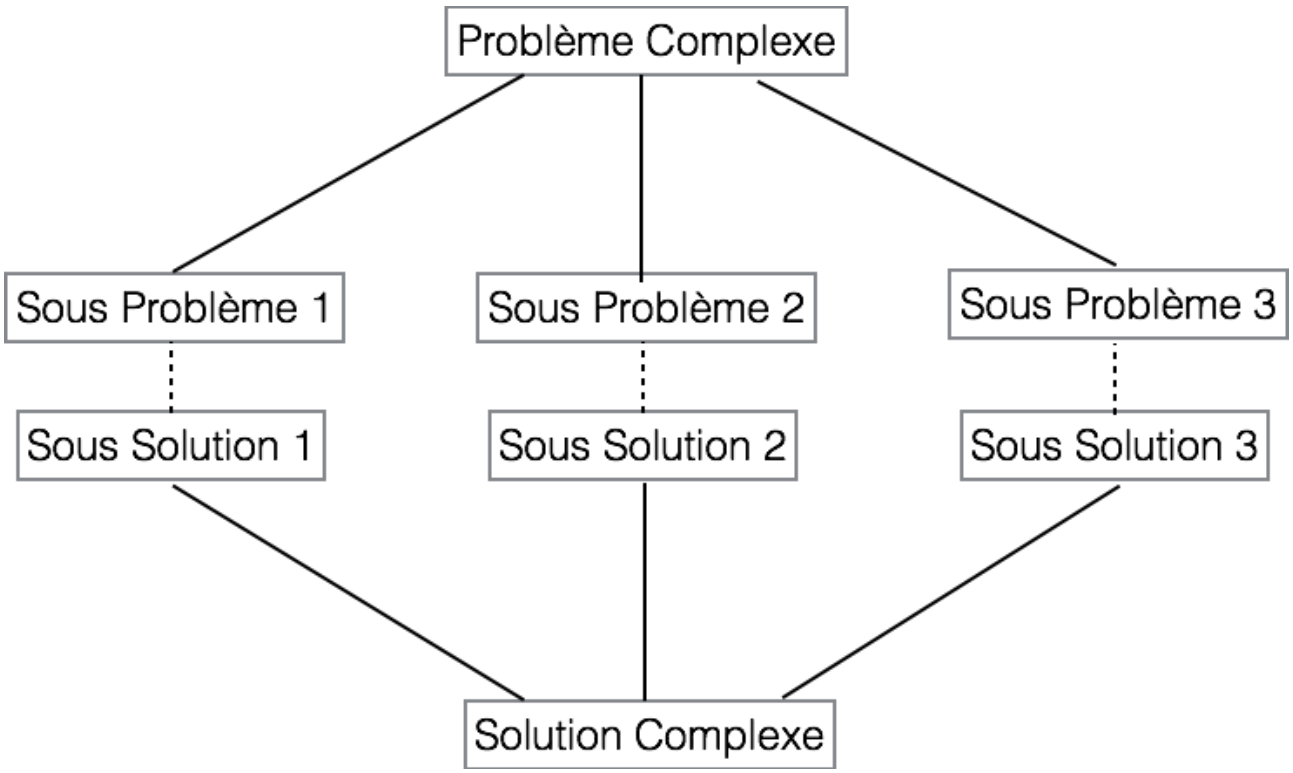


Figure N° 29: Décomposition d'un problème et La composition de sa solution

On définit une opération de base comme étant une opération connue en algorithmique telle que le tri d'un tableau. Une opération élémentaire, quant à elle, est une opération algorithmique simple (exemple : l'affectation).

Ainsi, au niveau 1, le problème est décomposé en un ensemble d'opérations de base, d'opérations élémentaires et d'opérations décomposables qui peuvent être liées par des structures de contrôle. Le nombre de niveaux de décomposition dépend de la complexité du problème à résoudre. En descendant dans les niveaux, seules les opérations décomposables sont décomposées, et cette décomposition s'arrête lorsqu'on arrive à un niveau constitué uniquement d'opérations de base et d'opérations élémentaires (figure 29).

Cette approche évite à l'apprenant de se noyer dans les détails dès le départ et diminue graduellement la complexité du problème abordé. En plus, l'apprenant peut librement exprimer sa solution, sans aucune influence ou restriction, ce qui favorise l'autonomie.

Notre objectif, par cette approche, est d'évaluer des solutions algorithmiques. Cependant, la retombée essentielle est l'apprentissage par l'apprenant de la décomposition. En effet, celle-ci est un passage obligé pour l'apprenant dans la formulation de sa solution.

1. Une méthode basée sur l'appariement structurel

L'appariement structurel détecte les correspondances en fonction de la structure des graphes.

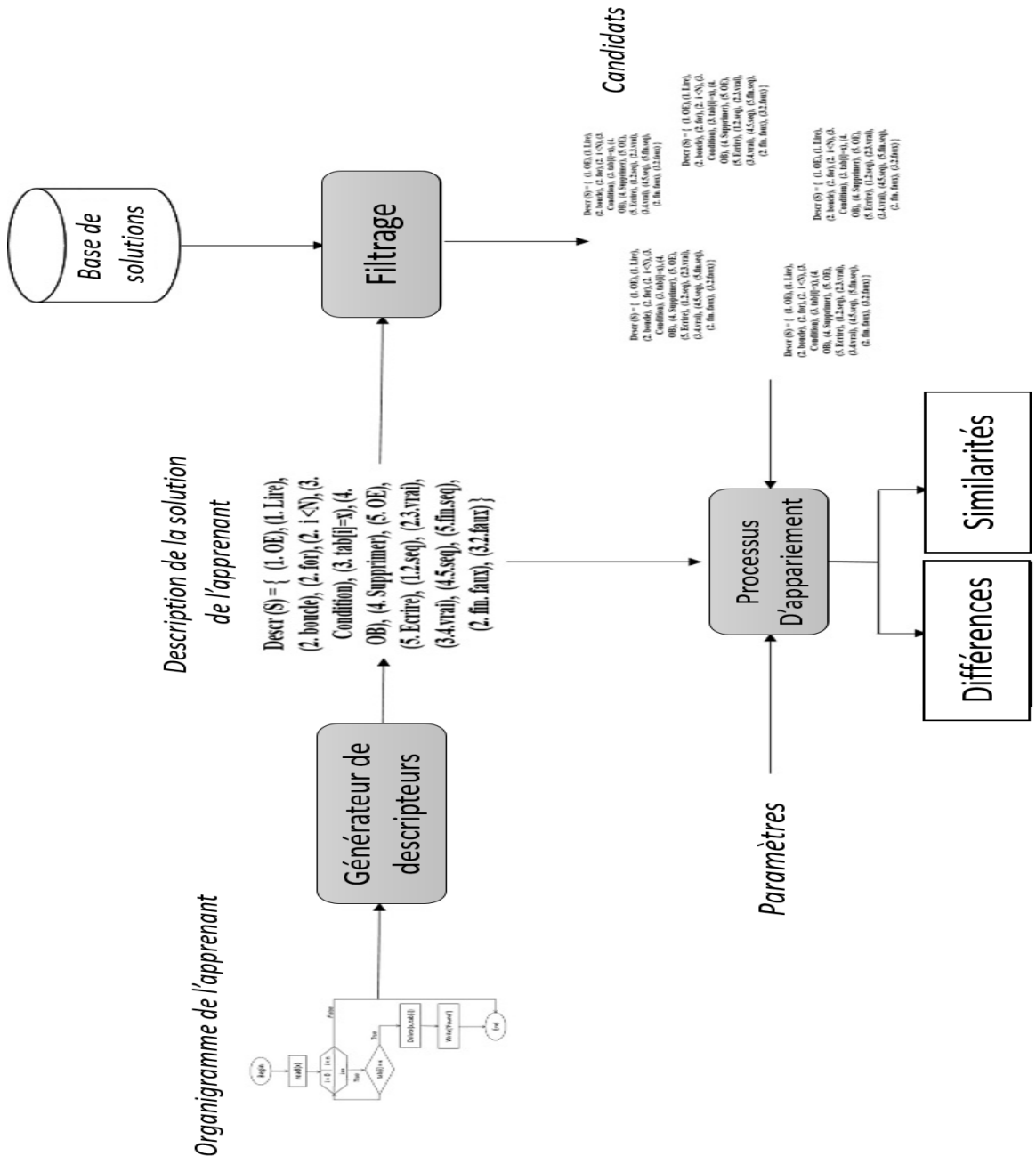
Anchor-PROMPT est une des premières méthodes d'appariement structurel utilisée pour aligner des ontologies du web sémantique, Noy et Musen (2001). Cette méthode prend en entrée un ensemble d'ancres (des correspondances exactes entre deux classes) et retourne un nouvel ensemble de correspondances entre classes. Cette méthode considère l'ontologie comme un graphe dans lequel les classes sont des nœuds du graphe et les propriétés des classes sont des arcs. Cette méthode analyse les chemins de même longueur entre deux ancres. Deux nœuds de deux chemins qui apparaissent dans la même position obtiennent un score non nul. Le score entre ces deux nœuds augmentera s'ils apparaissent à la même position dans deux autres chemins. Enfin, les correspondances obtenues en sortie sont les couples de classes ayant un score élevé. Cette méthode ne prend pas en compte l'étiquette des arcs (le nom des propriétés) entre les nœuds.

IV. Comment évaluer l'organigramme de l'apprenant ?

La figure ci-dessus représente une architecture fonctionnelle de la méthode proposée. Elle est composée principalement de trois composants essentiels : le générateur de descripteurs, le filtrage et le processus d'appariement (Aiouni et al. 2016).

IV.1. Généralités sur l'appariement

Avant de définir les différents problèmes et les techniques d'appariement classiques, il convient de préciser le sens du terme appariement (matching ou match). Le terme appariement prête à confusion car il peut être aussi bien un processus (ou une tâche) que le résultat de ce même processus. Les définitions de l'appariement varient sensiblement en fonction de ce facteur mais également en fonction de la nature des données à apparier, du domaine, du contexte d'application et de la présentation des résultats. Nous donnons ici quelques définitions que nous avons rencontrées au cours de nos lectures relatives au problème d'appariement et nous caractérisons le problème d'appariement de modèles.



L'appariement de formes ou de motifs (pattern matching) est la mise en correspondance de formes selon un ensemble prédéfini de règles ou de critères. L'appariement de formes se ramène à un problème de filtrage. Dans un programme informatique, le filtrage par motif (une autre traduction de pattern matching) est le fait de vérifier la présence des constituants d'un motif donné. À la différence de la reconnaissance de formes (pattern recognition), les motifs sont spécifiés rigidement et concernent conventionnellement des séquences ou des arbres. Le filtrage par motif est utilisé pour vérifier qu'un objet filtré a une structure désirée, pour trouver une structure appropriée, pour retrouver des parties alignées ou pour substituer les motifs reconnus par quelque chose d'autre.

Dans le cadre des ontologies, l'appariement (ontologymatching) est le processus de découverte des relations (ou correspondances) entre les entités de différentes ontologies (Euzenat&Shvaiko, 2007). Le terme alignement est employé plus communément dans le contexte des ontologies. L'alignement d'ontologies met en évidence les relations sémantiques de plusieurs ontologies à confronter (équivalence, subsomption, incompatibilité, etc.). L'expression des correspondances, appelée aussi alignement, peut par la suite être utilisée par exemple pour fusionner les ontologies, migrer des données entre ontologies ou traduire des requêtes formulées en fonction d'une ontologie vers une autre (DjoufakKengue et al., 2008).

L'appariement de schémas (schemamatching) consiste à trouver les correspondances sémantiques (i.e. appariements) entre deux schémas (Do et Rahm, 2007). L'appariement peut être considéré comme une opération ou un opérateur (match) qui prend deux schémas en entrée et produit un mapping entre les éléments des deux schémas correspondant sémantiquement les uns aux autres (Rahm et Bernstein, 2001).

Les problèmes d'appariement de graphes (graph matching) consistent à mettre en correspondance les sommets (noeuds) de deux graphes, l'objectif étant généralement de comparer les objets modélisés par les graphes.

Nous avons pu constater que la terminologie relative au problème d'appariement diffère entre les différents domaines et même au sein d'un même domaine. Quelles que soient la nature des modèles à analyser (ontologies, schémas, graphes) et leur représentation retenue en interne, la plupart des modèles en entrée peuvent être transcrits sous forme de graphes. Le but général du processus d'appariement est le même pour tous les types de modèles : l'identification et la qualification de correspondances entre les éléments.

Le résultat du processus d'appariement (match result), appelé également appariement, alignement ou mapping indique quels éléments des modèles en entrée correspondent les uns aux autres. Quelques variations d'interprétation existent entre les notions d'alignement et de mapping (Euzenat et Shvaiko, 2007). Un alignement est un ensemble de correspondances entre deux ou plusieurs (cas d'appariements multiples) modèles (par analogie aux alignements de séquence moléculaire). Une correspondance est une relation entre plusieurs éléments des modèles. Le terme mapping peut être considéré de manière équivalente à celui d'alignement. Néanmoins, dans le contexte des appariements de schémas, le terme alignement est très peu usité et celui de mapping est préféré. Cependant, un mapping peut aussi être vu comme une version orientée ou dirigée d'un alignement. La définition mathématique requerrait en principe que l'objet d'origine soit égal à son image (i.e. une relation d'équivalence). Un mapping est comme une collection de règles (mappingrules) toutes orientées dans la même direction, c'est-à-dire d'un modèle vers un autre, et telle que les éléments du modèle source apparaissent une fois au plus. Dans un mapping, une mappingrule associe un élément d'un modèle à un élément de l'autre modèle. Nous conservons la définition du mapping comme une version dirigée d'un alignement dans le reste du chapitre.

Le résultat final d'appariement peut associer un ou plusieurs éléments du premier modèle à un ou plusieurs éléments de l'autre modèle et réciproquement. Dans (Rahm et Bernstein, 2001) des relations de cardinalité entre les différents éléments appariés sont introduites pour qualifier les appariements produits. Elles correspondent à quatre cas de figure : 1:1 (un élément associé à un élément : couplage), 1:n (un élément associé à n éléments), n:1 (n éléments associés à un élément), ou m:n (m éléments associés à n éléments). Les trois derniers cas de figures correspondent à des appariements multiples (multivoques) d'éléments. De plus, un élément d'un modèle peut intervenir dans zéro, une ou plusieurs correspondances d'éléments de l'alignement produit entre deux modèles : la cardinalité est qualifiée de globale à cette échelle. En outre, à l'intérieur d'une correspondance individuelle, un ou plusieurs éléments d'un diagramme peuvent être appariés à un ou plusieurs éléments de l'autre modèle : la cardinalité de l'appariement est qualifiée ici de locale. Les cas de cardinalité globale qui concernent tous les éléments appariés sont orthogonaux aux cas des éléments appariés individuellement. C'est-à-dire qu'un résultat d'appariement peut avoir une cardinalité globale 1:n en combinaison avec des cardinalités locales d'appariement 1:1 ou 1:n.

La plupart des approches d'appariement sont restreintes à la cardinalité locale 1:1 car elles sélectionnent comme candidat d'appariement pour un élément d'un modèle, l'élément le

plus similaire dans l'autre modèle (Do et al., 2002). Les alignements produits ont donc en général une cardinalité globale 1:1 ou 1:n. Plus d'efforts et l'utilisation de critères plus sophistiqués sont nécessaires pour générer des appariements locaux et globaux de cardinalité n:1 et m:n. L'obtention d'alignements d'éléments de cardinalité globale m:n requiert par exemple de prendre en compte les structures agençant les éléments dans les modèles.

IV.1.1. Générateur de descripteurs

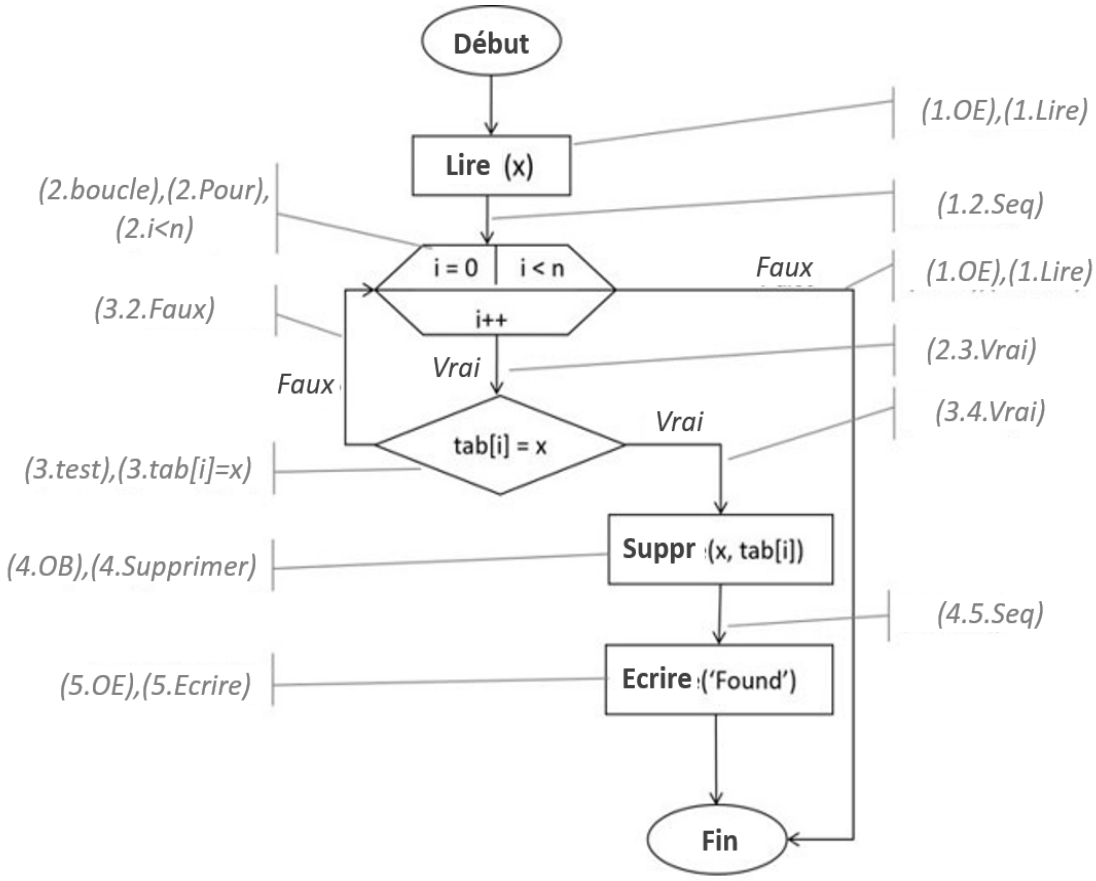
Pour automatiser la comparaison du processus d'apprentissage avec ceux de l'expert (plan de solutions), nous avons été inspirés par le travail de Sorlin (Sorlin et al., 2006) sur la mesure des graphiques multiples marqués. Cette approche nous a permis de proposer une méthode pour adapter des solutions algorithmiques.

De la structure organisationnelle de l'apprenant, une description de la solution est générée. Pour cela, nous avons attribué à chaque opération et chaque transition un ensemble d'étiquettes. L'ensemble du couple (Num_operation, étiquette) et triplets (Num_operarationS, Num_operationC, étiquette) sont des descripteurs et constituent la description de la solution. Le couple (Num_operation, étiquette) décrit les opérations de l'organigramme O et les triplets (Num_operationS, Num_operationC, étiquette) transition entre le fonctionnement (S et C sont pour la source et la cible).

Étant donné Lo un ensemble d'étiquettes d'opération et Lt un ensemble d'étiquettes de transition, un organigramme étiqueté est défini par une triplet S = <O, ro, rt> tel que :

- O est un ensemble fini d'opérations,
- ro⊆ O × Lo est une relation associant des étiquettes aux opérations (nœuds dans l'organigramme), à savoir ro est l'ensemble des couples (opi, l) de telle sorte que l'opération opi est marquée par l. Pour chaque nœud de l'organigramme, deux descripteurs sont affectés, l'un contient la nature du nœud, si elle est une opération, un test ou une boucle et le second contient l'étiquette. Par exemple, en figure 31, pour l'opération de lecture (x) deux descripteurs ont été assignés; (1.OE) mentionne le type de l'opération (opération élémentaire) et (1.read) pour l'étiquetage (nom de l'opération).
- • rt⊆ O × O × Lt est une relation associe les étiquettes aux arcs, à savoir rt est l'ensemble des triplets (opi, opj, l) tel que l'arc (opi, opj) est marqué par l. Étiquette en transition peut prendre deux valeurs seq pour la transition de séquencement ou Vrai / Faux quand il est un test de remplacement. La figure 31 résume le processus d'étiquetage d'un organigramme.

La description de la solution S est l'ensemble de toutes ses caractéristiques d'opération et de transitions Desc (S) = ro U rt.



Descr(S)={(1.OE),(1.Lire), (2.boucle),(2.Pour),(2.i<n), (1.2.Seq),
(1.OE),(1.Lire), (3.2.Faux), (3.test),(3.tab[i]=x), (4.OB),(4.Supprimer),
(2.3.Vrai), (3.4.Vrai), (5.OE),(5.Ecrire), (4.5.Seq)}

Figure N° 31: Descripteurs d'un organigramme

IV.2. Filtrage

L'étape filtrage se déroule avant le processus d'appariement afin d'optimiser le processus de recherche de l'organigramme le plus semblable. Elle consiste à chercher parmi les organigrammes prédéfinis ceux qui contiennent toutes les opérations de base critiques qui ont été prescrites par l'expert. Le filtrage permet de diminuer le nombre de solutions à mettre en correspondance. Par conséquent, on obtient un sous-ensemble de solutions provenant des prédéfinis qui contiennent des opérations critiques. Ce sous-ensemble de solutions sera présenté en tant que « candidats » où nous essayons de trouver la solution la plus proche de la solution de l'apprenant par le mécanisme d'appariement.

La validation d'une solution fournie par l'apprenant, se fait en comparant cette solution avec toutes les solutions de l'expert, pour le même exercice, référencées dans la base des plans.

IV.3. Processus de matching

Afin d'automatiser la comparaison de la démarche de l'apprenant avec celles de l'expert (plan de solutions) nous utilisons un appariement structurel que nous allons définir par la suite.

IV.3.1. Processus d'appariement

Le processus d'appariement peut être défini sous forme d'une « boite noire». Cette boite noire est une opération qui détermine l'alignement A' pour une paire de modèles m et m'. Quelques autres paramètres peuvent étendre la définition du processus : l'utilisation en entrée d'un alignement A qui est complété par le processus, des paramètres d'appariement (par exemple des poids et des seuils), des ressources externes utilisées par le processus d'appariement. Techniquement, le processus d'appariement peut être vu comme une fonction f qui, à partir d'une paire de modèles à apparier m et m', un alignement en entrée A, un ensemble de paramètres p et un ensemble de ressources externes r (par exemple des thésaurus), retourne un alignement A' entre ces modèles :

A' = f(m, m',A , p, r)

Le processus d'appariement peut être représenté schématiquement comme c'est le cas dans la figure suivante :

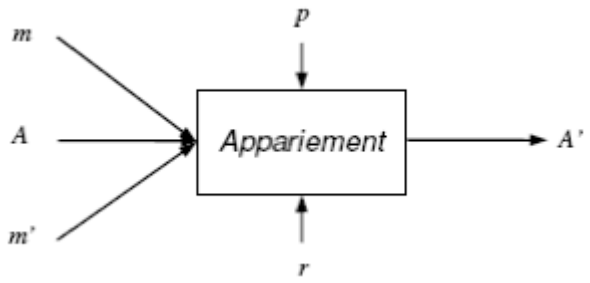


Figure N° 32: Processus d'appariement

Il peut être utile dans certains cas de considérer spécifiquement l'appariement de plus de deux modèles avec le même processus (Euzenat et Shvaiko, 2007). Ce processus est nommé dans ce cas appariement multiple (multiple matching). Le processus d'appariement multiple peut alors être vu comme une fonction f qui à partir d'un ensemble de modèles à apparier {m1, ...mn}, un alignement en entrée A, un ensemble de paramètres p et un ensemble de ressources, retourne un alignement A' entre ces modèles.

A' = f(m1, ...mn, A, p, r)

IV.3.2.Problèmes d'appariement et mesure de similarité

Nous venons de présenter le processus d'appariement sous forme d'une boite noire. En interne, l'appariement de plusieurs modèles et par conséquent de leurs constituants implique qu'ils soient comparés selon divers critères. Le problème de comparaison de plusieurs objets repose notamment sur l'évaluation de leurs similarités ou de leurs différences. La mesure de la similarité de deux objets consiste à identifier et à quantifier leurs points communs alors que la mesure de distance entre deux objets est l'identification et la quantification de leurs différences. Dans le contexte des graphes et donc des modèles assimilables à des graphes (c'est le cas pour les majorités des modèles manipulables dans les approches d'appariement de schémas ou d'ontologies), distance et similarité sont deux concepts duaux qui renvoient à un objectif commun. Le calcul de la distance ou de la similarité de deux structures de graphes permet habituellement de trouver le « meilleur » appariement des sommets d'un graphe, c'est-à-dire celui qui préserve le plus de caractéristiques des sommets et des arcs (Sorlin et al., 2007). Vu sous un autre angle, l'appariement est un problème d'optimisation où l'objectif est de trouver un alignement induisant la distance la plus faible entre les objets comparés.

Un appariement de graphes est dit univoque (univalent) lorsque chaque sommet d'un graphe est associé avec au plus un sommet de l'autre graphe (cardinalité d'appariement 1:1). Un appariement est dit multivoque lorsque chaque sommet d'un graphe peut être associé à un

ensemble de sommets de l'autre graphe (Sorlin et al., 2007). Un appariement multivoque d'un sommet d'un graphe avec plusieurs autres sommets de l'autre graphe se nomme généralement éclatement de sommets (cardinalité d'appariement 1:n). Inversement, un appariement multivoque de plusieurs sommets associés à un seul sommet de l'autre graphe est une fusion de sommets (cardinalité d'appariement n:1).

Différents problèmes courants d'appariement de graphes existent. Les mieux connus sont les appariements de graphes univoques exacts lorsque toutes les caractéristiques des sommets et des arcs sont préservées dans l'appariement ; c'est le cas notamment dans l'isomorphisme de graphes et de sous-graphes qui permettent respectivement de vérifier si deux graphes sont structurellement identiques (relation d'équivalence) ou si un graphe est « inclus » dans un autre (relation d'inclusion). Les appariements peuvent être univoques tolérants aux erreurs lorsque toutes les caractéristiques des sommets et des arcs ne peuvent pas être préservées lors de l'appariement (les objets appariés ne sont pas équivalents en tout point) : la recherche du plus grand sous-graphe commun (la plus grande partie commune à deux graphes) ou le calcul de la distance d'édition de graphes (le plus petit nombre d'opérations à effectuer pour transformer un graphe en un autre) en sont deux exemples.

Dans les cas plus complexes où une mise en correspondance univoque (un sommet avec zéro ou un sommet) ne suffit pas, il est nécessaire de mettre en place une mesure de similarité multivoque tolérante aux erreurs. Cette mesure est nécessaire dans les problèmes concernant la comparaison d'objets décrits à différents niveaux de granularité et où la recherche d'appariements multivoques (éclatements et fusions) est obligatoire. C'est le cas notamment dans des problèmes d'appariement d'une image bruitée à un modèle schématique (imagerie du cerveau humain par exemple) ou dans des problèmes d'appariements d'objets sur/sous segmentés. Les problèmes de recherche de composants complexes ou patterns dans des modèles en génie logiciel nécessitent également ce type de mesure pour restructurer ensuite les modèles de manière automatique.

IV.3.3. Mesure de similarité générique pour l'appariement de graphe

Nous allons présenter une mesure de similarité générique dans le contexte de l'appariement de graphes. Cette mesure de similarité nous a intéressés du fait de sa généricité.

Sorlin et Solnon ont proposé une mesure générique de distance paramétrable en fonction du type de graphes à apparier pour les différents problèmes d'appariement de

graphes (Sorlin , 2006) (Sorlin et al., 2007). Cette mesure de similarité de deux objets est fonction de leurs caractéristiques communes sur l'ensemble de toutes leurs caractéristiques.

Elle est générique dans le sens où elle modélise les mesures de distance et de similarité de graphes existantes. De plus, elle présente l'avantage d'être paramétrée par des fonctions de similarité pouvant être exprimées en fonction des connaissances propres au domaine, à l'application considérée ainsi qu'au type d'appariement recherché (univoque et/ou multivoque). De manière générale, deux fonctions de similarité permettent respectivement de calculer le score de similarité de chacun des sommets et des arcs.

Notre objectif est d'évaluer des productions d'apprenants (solutions algorithmiques). Pour cela, la solution à valider est comparée avec celles de l'expert.

Ce besoin de mesurer la similarité a pour but de faciliter et d'automatiser la validation. Notre motivation pour l'appariement structurel vient de la représentation formelle que peut avoir un algorithme, qui est l'organigramme.

Un premier point important de la mesure de similarité que nous utilisons par rapport aux mesures existantes, est qu'elle n'est pas seulement quantitative (évaluant le degré de similarité de deux solutions) mais aussi qualitative (explicitant en quoi les solutions sont similaires et en quoi elles sont différentes).

Un second point important de notre mesure est qu'elle permet de définir l'importance relative des caractéristiques, les unes par rapport aux autres et par conséquent d'introduire la connaissance dans le calcul de similarité.

Afin d'automatiser la comparaison de la démarche de l'apprenant avec celles de l'expert (plan de solutions), et en s'inspirant des travaux de Sorlin (2006) sur la mesure des graphes multi-étiquetés, nous proposons une méthode pour l'appariement de solutions algorithmiques.

Cette méthode consiste à comparer la solution de l'apprenant avec chacune des solutions de l'expert afin de mesurer la similarité entre elles. Elle est constituée de deux étapes séquentielles.

A partir de l'organigramme de l'apprenant, une description de la solution est générée. Pour cela, on affecte à chaque opération ainsi qu'à chaque transition un ensemble d'étiquettes. L'ensemble des couples (Num_opération, étiquette) et des tuples (Num_opérationS, Num_opérationC, étiquette) sont des descripteurs et constituent la description de la solution.

A un organigramme, on attribue une définition formelle S = < O, ro,rt>. Étant donné Lo un ensemble fini d'étiquettes d'opérations et Lt un ensemble fini d'étiquettes de transitions, tels que :

- O est un ensemble fini d'opérations.
- ro $\subseteq$ O * Lo est la relation associant opérations et étiquettes, i.e. ro est l'ensemble des couples (oi, l) tels que l'opération oi est étiquetée par l.
- rt $\subseteq$ O * O * Lt est la relation associant transitions et étiquettes, i.e. rt est l'ensemble des tuples (oi, oj, l) tels que la transition (oi, oj) est étiquetée par l.

La description de la démarche S est l'ensemble de toutes ses caractéristiques d'opérations et de transitions, Descr (S)=ro U rt.

Comme le montre la figure 4.7, l'étiquetage de l'organigramme de l'apprenant se fait de la manière suivante.

Pour les structures de contrôle :

- Une boucle est décrite par trois étiquettes, la première indique que c'est une boucle, la seconde son type et la troisième son intervalle.
- Une condition est décrite par deux étiquettes, la première indique que c'est une condition, la deuxième la condition elle-même.
- Les opérations sont décrites par deux étiquettes, la première montre le type de l'opération (opération de base ou élémentaire), la deuxième le nom de l'opération.

Les transitions n'ont qu'une seule étiquette qui peut être : vrai, faux ou seq (dans le cas d'un simple séquencement).

De la manière suivante la description d'une solution est générée. C'est la réunion de tous les couples décrivant les opérations (structures de contrôle inclus), ainsi que les tuples décrivant les transitions, la figure 4.8 montre la description de la solution de la figure 4.7.

IV.3.3.1. Le calcul de la similarité entre solutions

Un appariement entre deux démarches S1 = < O1, rO1, rt1 >et S2 = < O2, rO2, rt2 >, est une relation :

- m $\subseteq$ O1*O2.

Un tel appariement associe à chaque opération d'une solution l'opération du même ordre de l'autre solution.

Pour mesurer la similarité entre deux démarches par rapport à l'appariement m, nous proposons d'adapter la formule de similarité de Tvesky (1977) généralisée par sorlin (2006) :

$$Sim1_m(S1,S2) = \frac{f(descr(S1) \cap descr(S2))}{f(descr(S1) \cup descr(S2))} \qquad (1)$$

La formule (1) calcule la similarité de deux solutions, en mettant en correspondance leurs descriptions.

La fonction f définit l'importance relative des descripteurs, les uns par rapport aux autres. Cette fonction formule (2) est souvent définie comme une somme pondérée :

$$f(F) = \sum_{(o,l)\in F} poids(o,l) + \sum_{(o1,o2,l)\in S} poids(o1,o2,l) \qquad (2)$$

L'attribution des poids aux différents descripteurs d'une solution est réalisée par l'expert, celui-ci tient compte de l'importance du descripteur, ainsi que de l'objectif de l'exercice résolu par l'apprenant.

Nous allons présenter un exemple de calcul de similarité entre une solution d'apprenant et une solution d'expert référencée :

La figure 33 montrent respectivement l'organigramme de l'expert et sa description et l'organigramme de l'apprenant et sa description. En mettant leurs descriptions en correspondance, suivant l'appariement m, on remarque dans la figure 32 les descripteurs communs des deux solutions en rouge.

Il est évident que la solution retenue sera celle présentant la plus grande similarité. Celle-ci doit impérativement être inférieure à un seuil d'acceptabilité indiqué par l'expert. La note attribuée dépend du degré de similarité maximal entre la solution à valider et les solutions de l'expert.

Lorsque le degré de similarité est inférieur au seuil, la solution est envoyée à l'expert pour évaluation. Si celui-ci la juge intéressante, la solution sera ajoutée au plan de solutions de l'exercice, ce qui assure l'évolutivité de la base des plans, dans le cas contraire la solution de l'apprenant est rejetée.

Cette évolutivité du plan de solutions garantit une évaluation quel que soit la solution proposée par l'apprenant. Ainsi, plus le temps passe plus l'évaluation différée diminuera au profit de l'évaluation directe.

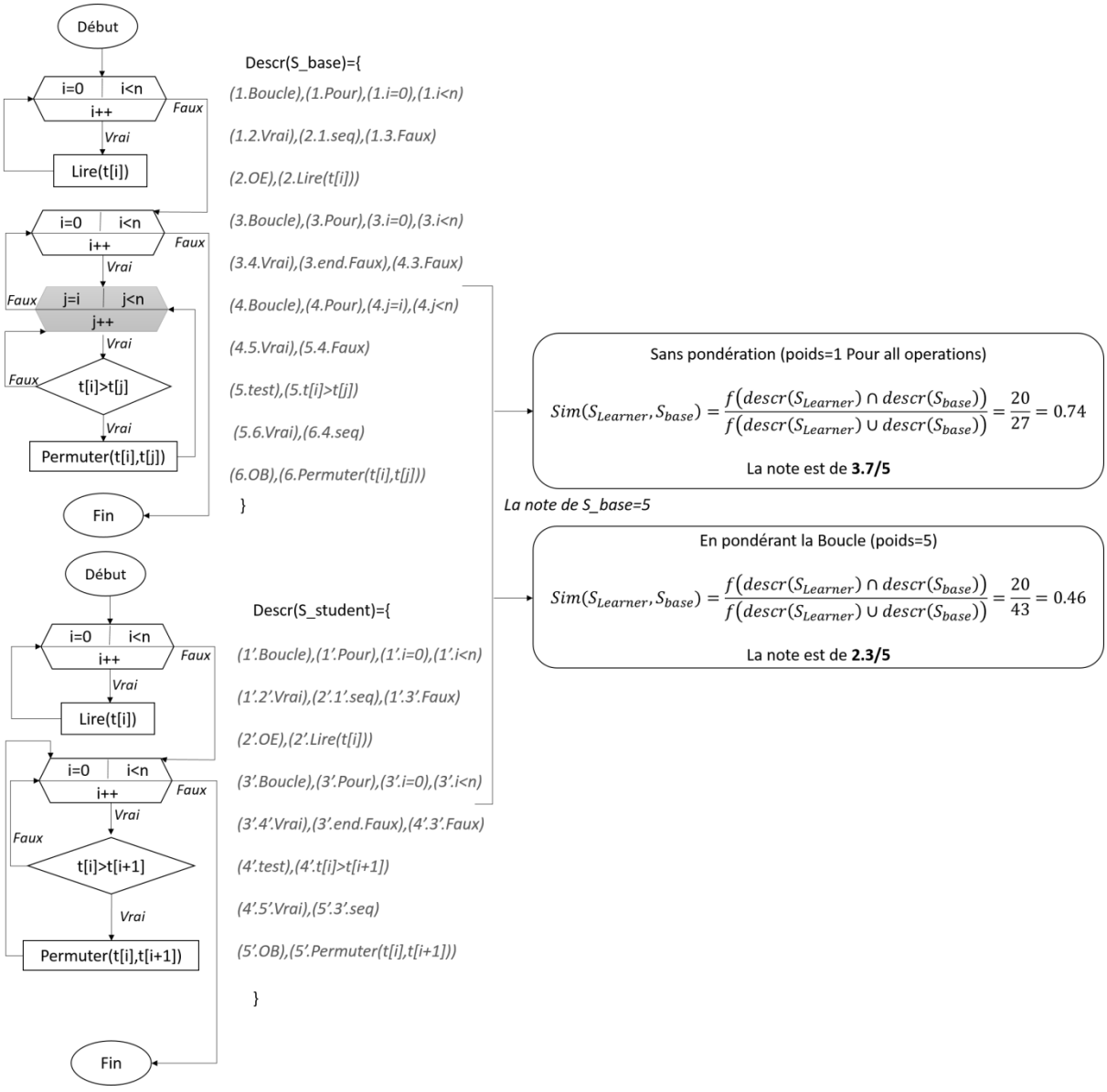


Figure N° 33: Processus de matching

V. Conclusion

L'approche proposée dans cette thèse a pour objectif d'évaluer les organigrammes des apprenants en algorithmique. Cette méthode est basée sur l'appariement structurel. L'évaluation exploite des aspects structurels et sémantiques intrinsèques à ces modèles. Leur comparaison, reposant sur des fonctions de similarité paramétrables, peut s'adapter aux modèles et au type d'appariement recherché.

Après avoir défini la méthode, nous l'avons implémentée et testée. Dans le chapitre suivant, nous allons présenter eAlgo, un outil d'évaluation automatique des organigrammes en algorithmique pour les apprenants novices ainsi que son expérimentation.