Analyse causale de performance des algorithmes à partir de données d'observation

Ahmed Mabrouk
Faculté des sciences
Université de Montpellier II
Email : Ahmed.Mabrouk@etud.univ-montp2.fr

Jan Lemeire
Electronics and Informatics Department
Vrije Universiteit Brussel
Email : jan.lemeire@vub.ac.be

Rune Erlend Jensen
Institutt for datateknikk
og informasjonsvitenskap
Norges teknisk-naturvitenskapelige universitet
Email : runeerle@stud.ntnu.no

***Résumé*—L'analyse de performance des algorithmes en fonction des paramètres fait actuellement l'objet de nombreux travaux. La tâche est ardue, car il s'agit d'étudier la relation entre les paramètres et la performance à partir de très grandes masses de données d'observation. Nous proposons dans cet article un nouvel outil d'analyse de performance basé sur l'inférence causale, en faisant l'apprentissage des modèles causaux de performance basés sur les réseaux bayésiens causaux étendus de type multi-nets (BMN) afin de capturer des informations supplémentaires concernant les contextes spécifiques d'indépendances (CSI) et les outliers. Nous présentons les extensions de l'algorithme d'apprentissage PC qui mettent en application ces idées et qui tiennent compte également de la coexistence des variables discrètes et continues et de la possibilité d'existence des relations déterministes entre eux. Nous montrons ensuite l'intérêt et l'utilité des BMN dans l'analyse de performance à travers les conclusions claires qu'ils offrent aux développeurs.**

Keywords : analyse de performance des algorithmes, réseaux bayésiens causaux, multi-nets, mesures statistiques, apprentissage, structure.

I. INTRODUCTION

La nécessité d'analyser les performances des algorithmes est aujourd'hui reconnue et devient de plus en plus pressante et cruciale pour les développeurs. Étant donné que l'objectif initial de l'informatique est de fournir des algorithmes permettant de résoudre un problème donné, le développeur doit tout d'abord penser à développer des choses qui soient compatibles avec ces contraintes matérielles et ces besoins. C'est pour cela que la conception et l'implémentation des applications de hautes performances nécessite la connaissance de plusieurs facteurs qui peuvent influencer la performance. Le cas que nous allons traiter dans l'article consiste à chercher le compilateur et ses options qui donnent la meilleure performance pour un algorithme qui calcule la trajectoire des particules en appliquant un filtre Kalman. Le filtre appliqué comporte plusieurs opérations dont la multiplication des trois matrices qui fait l'objet de l'étude de performance de Rune [9].

Plusieurs méthodes statistiques ont été mises en place afin de supporter l'analyse de performance des algorithmes. Ces méthodes sont certes efficaces mais leur utilisation concrète demeure encore difficile ou limitée à des domaines restreints et elles sont vite dépassées par la quantité énorme de données. Parmi ces méthodes statistiques, on peut citer l'analyse de corrélation qui est destinée à quantifier et tester la liaison entre deux variables quantitatives, ou aussi l'analyse de régression qui est basée sur l'étude de la corrélation entre variables et qui est souvent utilisée comme outil au service de la prédiction. Étant donné que le but principal du développeur ne se contente pas des simples mesures statistiques ou des courbes, mais plutôt à avoir des explications plus détaillées afin de mieux comprendre les causes et les origines des résultats de performances, et par la suite une meilleure analyse de son algorithme. Avec les méthodes classiques il ne pourra jamais atteindre tous ces détails.

Donc l'objectif de notre projet est de faciliter la tâche d'analyse en proposant un outil efficace qui offre des modèles multifonctionnels. Les modèles employés dans notre cas se basent principalement sur les réseaux bayésiens causaux (RBc) qui sont une extension des réseaux bayésiens classiques où toute relation entre variable correspond à une relation causale [6] ce qui permet par la suite de faire des inférences causales. Nous proposons aussi une extension au niveau de la représentation graphique des RBc par l'ajout d'autres informations sur les arcs qui relient les nœuds. Ces informations traitent ce qu'on appelle le contexte spécifique des indépendances [2] (CSI) qui est obtenu par les réseaux bayésiens de type multi-nets modélisant les corrélations propres à chaque configuration du contexte choisi, et les outliers. Ces améliorations permettent de mieux détailler les relations causales et donnent une vision plus claire que les simples dépendances entre les variables fournies par les RBc classiques.

Le présent article est organisé comme suit : Après avoir présenté quelques définitions à propos des réseaux bayésiens, nous évoquerons le principe de l'analyse de performance des algorithmes et l'utilisation des RBc étendus pour ces fins. Nous décrirons ensuite les extensions de l'algorithme d'apprentissage PC [10] qui mettent en évidence la complexité des données et les idées d'analyse de performance des algorithmes. Enfin, nous terminons avec une série de résultats afin de tirer les premières conclusions de notre travail.

II. Généralités sur les modèles graphiques probabilistes

A. Les réseaux bayésiens

Un réseau bayésien [7] est un modèle graphique représentant la distribution de la probabilité jointe (JPD) $P(X)$ sur un ensemble de variables aléatoires $X = \{X_1, ..., X_n\}$ qui définissent des probabilités $P \in$ [0,1] pour chaque état possible $(x_1, \ldots, x_n) \in X_{1,dom}, ..., X_{1,dom}$ où $X_{i,dom}$ est le domaine de définition de chaque variable X_i. Il s'agit d'un graphe acyclique orienté, dans lequel les nœuds représentent les variables aléatoires, et les arcs symbolisent les relations existantes entre ces variables. La JPD sur l'ensemble des variables X de ce modèle s'écrit de la façon suivante :

$$P(X_1, ..., X_n) = \prod_{1..n} (P(X_i|pa(X_i))) \quad (1)$$

B. Les réseaux bayésiens causaux

Un réseau bayésien causal (RBc) est un réseau bayésien mais avec quelques propriétés supplémentaires [6]. La propriété principale réside dans le fait que chaque ensemble d'arcs $Pa(X_i) \rightarrow X_i$ ne représente plus seulement une dépendance probabiliste, mais aussi une relation causale. Dans un RBc, chaque CPD $P(X_i|pa(X_i))$ représente un processus stochastique dans lequel les valeurs de X_i sont choisies en fonction des valeurs de $Pa(X_i)$ (mais pas l'inverse).
En plus de leur aspect plus intuitif, les RBc permettent de faire des calculs d'inférence probabiliste classique (comme les RB usuels), mais aussi d'inférence causale, i.e. de calculer l'effet d'une intervention sur une certaine variable sur d'autres variables [8].

III. L'analyse de performance des algorithmes

La tâche d'analyse de performance des algorithmes nécessite des réponses détaillées sur les points mentionnés ci-dessous.

A. La détection des dépendances et les CSI entre les variables d'un algorithme

La découverte d'une dépendance entre les variables X et Y consiste à savoir s'il y a une influence entre ces deux variables et la quantifier. Cette information est de grande importance pour l'analyse de performance des algorithmes car elle permet de découvrir les différentes relations de dépendance entre les variables de modèle, ce qui permet de donner des explications pour la variation des performances de temps à autre. Un exemple de dépendance est donné par la courbe de figure 1 qui suit une forme superlinéaire entre les paramètres size et unhalted_core_cycles (temps d'exécution) et Olevel (le niveau d'optimisation choisi pour un compilateur). Donc avec cette information de dépendance le développeur peut expliquer l'augmentation de temps d'exécution (Unhalted_core_cycles) par l'augmentation de la taille des données (size). Une autre chose que le développeur peut détecter à partir de cette courbe, est ce qu'on appelle le contexte spécifique des indépendances [2] (CSI), où il s'agit d'une spécialisation du terme de dépendance. En regardant l'allure de

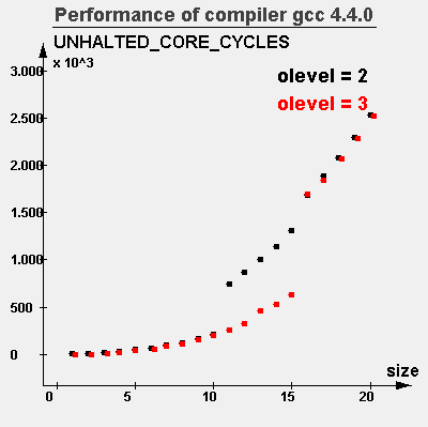

FIGURE 1. Analyse de dépendance entre les deux variables size, unhalted_core_cycles et Olevel.

la courbe 1, on peut conclure que les deux variables Olevel et Unhalted_Core_Cycles ne sont pas dépendantes au début, mais à partir d'un size ∈ [12,16] on voit qu'elles sont dépendantes.

B. La découverte des variables intermédiaires

L'un des buts de l'analyse de performance est de connaître les variables intermédiaires. Certains paramètres n'exercent pas d'influence directe sur la performance de l'algorithme, mais ils le font par le biais d'autre(s) variable(s). On dit ici que le paramètre X_i influence la performance à travers la variable X_k. Une telle relation de dépendance peut être modélisée comme suit :

$$X_i \rightarrow X_k \rightarrow performance$$

Dans les données d'analyse de compilateur, on remarque l'existence d'une telle relation, par exemple : le cas du paramètre size qui influence la performance (Unhalted_Core_Cycle) à travers la variable Instruction_retired qui est le nombre d'instructions réellement effectuées par le pipeline du processeur. La connaissance de ces variables est de grande importance pour le développeur lors de la phase d'analyse de son algorithme.

C. Détection des outliers

Ce sont des observations qui, pour une variable donnée, présentent des valeurs inhabituelles vu la distribution. L'objectif de la détection des points aberrants et influents est de repérer des points qui jouent un rôle anormal dans la régression jusqu'à en fausser les résultats. Supposons qu'on a deux paramètres $size_i$ et $size_{i+1}$ où $size_i < size_{i+1}$ et que ces deux paramètres donnent respectivement deux performances différentes $perf_i$ et $perf_{i+1}$ et que $perf_i > perf_{i+1}$. Ici le développeur est amené à trouver une explication pour cette exception qui est très mal reconstituée par la régression, et qui n'obéit pas de manière ostensible à la relation modélisée entre ces deux paramètres.

IV. Réseaux bayésiens causaux étendus pour l'analyse de performance

Un des problèmes communs lors de la tentative d'analyse d'un code est de savoir les raisons du changement de performance. Bien que les méthodes statistiques ordinaires permettent de donner des analyses et des explications pour une telle variation de performance, leur utilisation requiert vraiment des à priori en statistique. De plus, ces méthodes ne sont pas facilement manipulables et compréhensibles par tous les développeurs. Afin d'avoir un feedback rapide, détaillé et très précis, nous proposons un outil d'analyse de performance basé sur les réseaux bayésiens causaux étendus permettant de faire face à tous ces problèmes à travers une représentation concise et claire des variables et les relations entre eux, ce qui permet par la suite de faire des analyses efficaces à travers les inférences causales. Nous commençons tout d'abord à décrire comment un réseau bayésien causal étendu peut représenter les situations citées au niveau de la section 3 en nous appuyant sur des cas concrets tirés à partir des données d'analyse de compilateur. Puis nous présentons les extensions de l'algorithme d'apprentissage employées pour supporter une telle analyse.

A. La représentation des dépendances entre les variables

Si on regarde la définition des réseaux bayésiens causaux mentionnée au niveau de la section 3.A, on voit que la propriété principale de ces modèles réside dans le fait que chaque ensemble d'arcs $Pa(X_i) \rightarrow X_i$ ne représente plus seulement une dépendance probabiliste, mais aussi une relation causale. En partant de cette définition, les relations de dépendance entre les variables de la figure 1 peuvent être facilement représentées par un RBc comme on le voit au niveau de la figure 2. Une telle représentation est facilement compréhensible par l'analyseur et ne nécessite pas de connaissances en statistique. L'interprétation d'une telle figure est simple, c'est de dire que les deux variables Size et Olevel influent sur la variable unhalted_cycle_core qui est la performance dans notre cas.

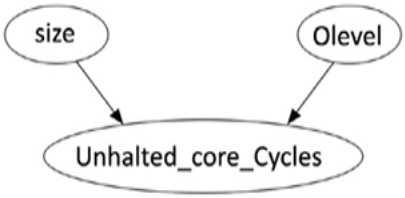

FIGURE 2. Représentation des dépendances entre les variables.

B. La représentation des CSI et les outliers

Les CSI [2] sont des propriétés très utiles pour une analyse de performance. Une telle propriété peut être modélisée par les RBc étendus dans le but de mieux personnaliser et détailler le graphe en lui rajoutant ces informations supplémentaires. Prenons l'exemple de la courbe (figure 1) qui est interprétée comme suit : En plus de la relation de dépendance entre ces trois variables, on remarque que les deux variables Olevel et Unhalted_Core_Cycles ne sont dépendantes qu'à partir d'un size $\in$[12,16]. Une telle information (CSI) peut être représentée facilement par un RBc étendu comme on le voit dans la figure 3. De la même façon les outliers peuvent être

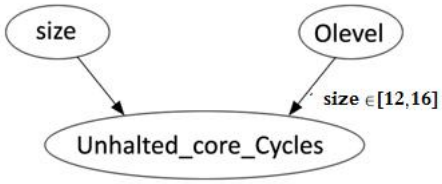

FIGURE 3. Représentation d'un contexte spécifique d'indépendance.

représentés par un RBc étendu par l'ajout des informations sur le(s) point(s) qui présentent ces exceptions (outliers) lors d'une dépendance entre deux variables continues. La figure 4 présente un outlier au niveau de la dépendance entre les deux variables size et Unhalted_core_cycles au point 16 (de size).

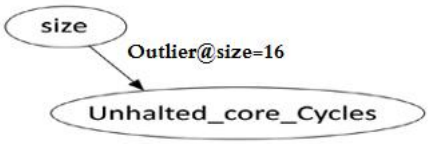

FIGURE 4. Représentation des outliers entre deux variables.

C. La représentation des variables intermédiaires

La découverte des variables intermédiaires est une information par défaut donnée par le RBc à travers sa représentation graphique. Exemple d'une telle variable est représenté dans la figure 5. Ce graphe peut être interprété comme suit : la variable size influence sur la performance à travers Instruction_Retired qui est une variable intermédiaire.

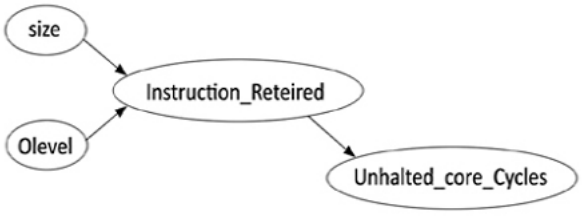

FIGURE 5. Représentation des variables intermédiaires.

V. Apprentissage de structures

A. Apprentissage d'un réseau bayésien

Dans cette section, nous présentons le fonctionnement élémentaire des algorithmes d'apprentissages basés sur des tests d'indépendance conditionnelle qui sont souvent appelés des méthodes de recherches sous contraintes telles que SGS, PC [10]. Toutes ces méthodes font l'hypothèse de suffisance

causale c'est à dire un ensemble de variables X est suffisant causalement pour une population donnée D si et seulement si dans cette population, chaque cause Y commune à plusieurs variables de X appartient aussi à X, ou si Y est constant pour toute la population.
Ces algorithmes se décomposent en trois phases :

1) Recherche d'indépendances conditionnelles dans les données en utilisant des tests statistiques.
2) Utilisation des indépendances découvertes pour former un graphe sans circuit partiellement dirigé (PDAG), en deux étapes :
   – les arêtes X–Y d'un graphe non dirigé complètement connecté sont supprimées pour chaque indépendance découverte ($X \perp\!\!\!\perp Y$).
   – le graphe non dirigé obtenu est ensuite partiellement dirigé grâce aux indépendances conditionnelles découvertes (elles correspondent à des V-structures $X \rightarrow Y \leftarrow Z$).
3) le PDAG obtenu est ensuite "completé" (CPDAG) grâce aux règles suivantes. Ce CPDAG est le représentant de tous les graphes équivalents au sens de Markov. Les arcs ajoutés lors de cette phase seront appelés par la suite arcs inférés.
   – si $X \rightarrow Y$, avec Z adjacent à Y mais pas à X, et que Y–Z n'est pas orienté, alors il faut l'orienter en $Y \rightarrow Z$.
   – s'il existe un chemin dirigé de X vers Y mais que X–Y n'est pas orienté, alors il faut l'orienter en $X \rightarrow Y$ pour ne pas créer de cycle.

B. Fidélité

Un réseau bayésien n'est pas capable de représenter n'importe quelle distribution de probabilité (ou la liste des indépendances conditionnelles associées). La première hypothèse que nous poserons est donc l'existence d'un réseau bayésien qui soit la P-map du modèle d'indépendance associé à la distribution de probabilité P sous-jacente à nos données. Cette hypothèse se retrouve souvent sous le terme de fidélité (faithfulness) entre le graphe et P.
Nous dirons donc qu'un graphe est fidèle à une distribution de probabilité lorsque toute indépendance probabiliste existant entre les variables considérées peut être déduites de la condition de Markov.

VI. Extensions

Vu la complexité que nous avons dénotée au niveau des données, nous pensons que l'approche d'apprentissage basée sur l'algorithme PC doit être étendue afin de tenir compte de tous les points mentionnés ci-dessous :

1) La base d'observation qu'on est en train d'analyser (analyse de compilateur) est caractérisée par la coexistence de variables discrètes (comme olevel qui prend soit 1 soit 0) et continues (comme size, unhalted_core_cycles).
2) L'existence des relations déterministes entres les variables, une telle relation permet d'exprimer une variable en fonction d'un ensemble de variables, ce qui va impliquer une détection supplémentaire des indépendances conditionnelles qui ne peuvent pas être représentées par des modèles fidèles.
3) Afficher les informations supplémentaires telles que les outliers et les CSI qui se rattachent aux réseaux bayésiens étendus et qui n'existent pas par défaut, afin d'assurer une analyse plus détaillée et efficace que par les simples RBc.

A. Mesure des dépendances entre variables continues et discrètes

La base de données « analyse de compilateur » est caractérisée par la coexistence de variables discrètes et continues. L'outil TETRAD que nous sommes entrain d'utiliser pour implémenter notre algorithme emploie le coefficient de corrélation Pearson pour les variables continues et χ_2 pour les variables discrètes[12] pour détecter les relations de dépendance. Un test statistique qui mesure le coefficient de dépendance entre deux variables dont l'une est continue et l'autre est discrète est inexistant. Afin de faire face à ce problème, nous avons employé l'information mutuelle qui calcule le degré de dépendance d'un couple de variables (X, Y) au sens probabiliste. L'information mutuelle mesure la quantité d'information apportée en moyenne par une réalisation de X sur les probabilités de réalisation de Y. En considérant qu'une distribution de probabilité représente notre connaissance sur un phénomène aléatoire, on mesure l'absence d'information par l'entropie $H(x)$[11] de cette distribution. En ces termes, l'information mutuelle s'exprime par :

$$I(X;Y) = H(X)H(X|Y) \quad (2)$$

$$H(X) = \sum_{x \in A} p(x) log p(x) \quad (3)$$

L'entropie conditionnelle est calculée comme suit :

$$H(X|Y) = \sum_{y \in B} H(X|Y = y) \quad (4)$$

I(X ;Y) est égale à zéro signifie que les deux variables X et Y sont indépendantes. et, dans le cas continu :

$$I(X,Y) = \int\int p(x,y) log \frac{p(x,y)}{p(x)p(y)} dxdy \quad (5)$$

où $p(x, y)$, $p(x)$ et $p(y)$ sont respectivement les densités des lois de (X, Y), X et Y.

B. Méthode de détection des outliers

L'utilisation des méthodes basées sur l'analyse de régression linéaire simple n'est pas efficace dans notre cas où plusieurs outliers peuvent être détectés. Pour cette raison, nous avons pensé à une autre méthode basée sur la régression multiple appelée distance de COOK [1], qui est souvent utile pour détecter les outliers multivariés (observations qui sont très

mal expliquées par le modèle). La première formulation de la distance de Cook D_i est la suivante [1] :

$$D_i = \frac{\sum_{j=1}^{n}[y_i - y_i(-i)]^2}{pMSE} \quad (6)$$

où MSE est la moyenne des erreurs du modèle de régression et p le nombre de coefficients du modèle. Cette méthode consiste à évaluer l'influence du point i sur la régression, en le supprimant du calcul des coefficients, et en comparant les prédictions avec le modèle complet (construit avec tous les points) et le modèle à évaluer (construit sans le point i). Si la différence est élevée, le point joue un rôle important dans l'estimation des coefficients. Il nous faut définir la valeur seuil à partir de laquelle nous pouvons dire que l'influence est exagérée. La règle la plus simple est : Que lorsque $D_i > 1$ le point est considéré comme outliers.

C. Multi-nets pour la détection des CSI

La méthode employée pour découvrir les CSI est basée sur l'approche multi-nets [2] qui consiste à apprendre un réseau bayésien de type multi-nets BMN (figure 6) modélisant les corrélations propres à une configuration (ou état) étant donné une troisième variable appelée contexte. Par conséquent, la structure d'un BMN peut signifier que deux ensembles de variables sont indépendantes étant donné une certaine configuration d'une troisième variable (contexte), et dépendantes étant donné une autre configuration de cette troisième variable. Un BMN est défini par une distribution de probabilité pour H et un ensemble des BN pour $X \setminus \{H\}$ (dans le cas discret), où chacun de ces derniers (BN) encode la distribution de probabilité jointe $P(X|H = h_i)$. L'algorithme d'apprentissage du BMN est le suivant :

Algorithm 1 Détection des CSI

$tabBN[] \leftarrow \emptyset$
$H \leftarrow \mathrm{X}_i$
if $H \in \mathbb{R}$ **then**
  $m \leftarrow \frac{|H|}{Taille\_intervalle}$
  soit h_m le window m associé à H
else
  soit h_m la $m^{éme}$ valeur prise par H
end if
for chaque configuration $\mathrm{h}_m \in H$ **do**
  if $H \in \mathbb{R}$ **then**
    $D^{'} = \forall d_i \in D : h_i \in h_m$
  else
    $D^{'} = \forall d_i \in D : h_i = h_m$
  end if
  tabBN()$\leftarrow$ apprendre_BN($D^{'}$)
$m \leftarrow$m+1
end for
Combiner les réseaux bayésiens de tabBn : $\forall\ X_i - X_j$ (i$\neq$ j) dans un contexte h_m et X_i est séparée de X_j dans un contexte h_k $(k \neq m)$, alors mentionner h_m sur l'arc $X_i - X_j$

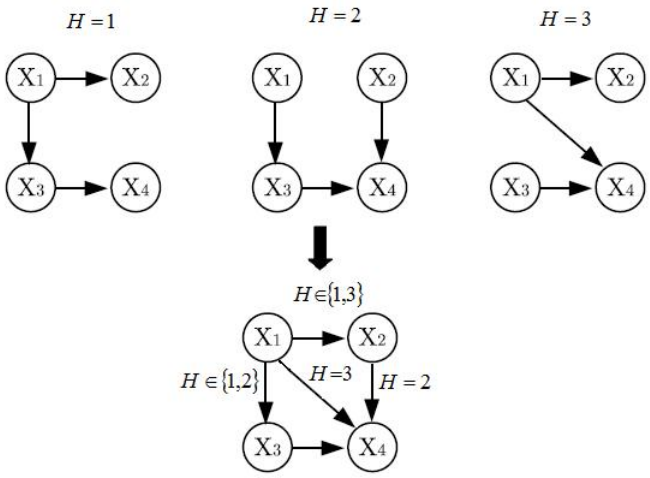

FIGURE 6. Exemple d'approche par multi-nets pour une variable de contexte (discrète) H à trois valeurs et la combinaison des RB obtenus.

D. Apprentissage des réseaux bayésiens causaux

Considérons cette structure $Z \rightarrow X \rightarrow Y$ et la relation déterministe $X = f(Z)$. Les deux indépendances conditionnelles qui peuvent résulter de cette relation sont :

$$Z \perp\!\!\!\perp Y|X \& X \perp\!\!\!\perp Y|Z$$

Nous appelons Y et Z des informations équivalentes à l'égard de X [3,5]. La première équation est issue de la condition de Markov, tandis que la seconde est impliquée par la relation fonctionnelle. La variable X est complètement déterminée par Z, donc Z a toutes les informations sur X.
Dans l'exemple, nous avons $[X \perp\!\!\!\perp Y|Z]$, ce qui est incorrect de considérer que Z sépare X de Y et l'arc entre X et Y peut être enlevé. Aussi $[Z \perp\!\!\!\perp Y|X]$ suggère que l'arc entre Y et Z peut être enlevé. Comme conséquence de ces deux actions, on va avoir un DAG non Markovien. L'algorithme utilisé VCP [4](very conservative PC algorithm) tient compte de l'existence de ces relations déterministes et permet d'apprendre une classe d'équivalence correcte.

VII. OBSERVATIONS ET LIMITES

Le tableau 1 présente les différentes mesures de dépendances entre les paramètres Olevel et unhalted_core_cycle pour les trois bases d'observations :
- gcc4.4.0 comporte les données issues de la version 4.4.0 du compilateur gcc ;
- gcc4.x.x comporte les données issues de la version 4.4.0, 4..2.4 et 4.3.3 du compilateur gcc ;
- gcc&icc comporte les données d'observation issues des deux compilateurs icc et gcc.

D'après ces résultats, on remarque que l'algorithme d'apprentissage a surmonté le problème de coexistence des variables discrètes et continues à travers l'utilisation de l'information mutuelle. Cependant, d'autres limites nous paraissent importantes à dépasser concernant la performance de test statistique utilisé qui dépend fortement de la taille des

données. D'après la première ligne du tableau 1, on remarque que la valeur du test statistique baisse en fonction de la taille des données observées de 0,047 pour gcc4.4.0 à 0,09 pour gcc&icc qui comporte plus de données. Cette diminution est expliquée par la forte existence de dépendances contextuelles qui interrompent le bon fonctionnement des tests statistiques lorsqu'il y a beaucoup de données. Le conditionnement sur des paramètres principaux, comme on le voit au niveau des lignes 2 et 3 du tableau 1, diminue l'influence des dépendances contextuelles et permet par la suite de détecter la dépendance (fixée à un seuil de 0,05) entre les deux paramètres Olevel et Unhalted_Core_Cycles.
Le même problème est rencontré lors de la détection des outliers et des CSI avec les méthodes d'analyse de régression multiple (distance de Cook) et l'algorithme de détection des CSI qui ne réussissent pas à renvoyer les bons résultats au cas où il y a beaucoup de données.
Donc pour conclure, nous pouvons dire que pour avoir des résultats beaucoup plus précis il y a deux choix possibles. Le premier consiste à filtrer les données d'observation et travailler d'une façon incrémentale afin d'échapper au problème de dépendance contextuelle. Alors que pour le deuxième choix, on conditionne sur des paramètres principaux pour avoir beaucoup plus de précision.

VIII. Conclusion

Dans cet article, nous avons abordé le problème d'analyse automatique de performance des algorithmes à partir des données expérimentales. Nous avons aussi traité le problème d'apprentissage des réseaux bayésiens causaux étendus à partir de données d'observation, qui vont être utilisés comme des modèles pour supporter une telle analyse à travers une représentation facilement compréhensible par l'analyseur qui ne nécessite pas de connaissances en statistique.
Nous avons présenté en outre les différentes extensions ainsi que les méthodes nécessaires pour mettre en place un modèle causal robuste. Les méthodes employées tiennent compte de la complexité des données dans les cas réels (les données d'analyse de compilateur) et permettent de mieux détailler les relations de dépendances entre les variables à travers l'ajout des informations supplémentaires concernant les outliers et les contextes spécifiques des indépendances par le biais des BMN.
Le sujet traité ouvre de nouvelles perspectives concernant le problème de contextualité qui présente un obstacle face à l'efficacité des tests statistiques, ce qui fera l'objet de nos futurs travaux.

	gcc 4.4.0	gcc 4.x.x	gcc & icc
$I(olevel; UNHALTED\_CORE\_CYCLES)$	0.047	0.025	0.009
$I(olevel; UNHALTED\_CORE\_CYCLES \mid size)$	**0.182**	**0.115**	0.020
$I(olevel; UNHALTED\_CORE\_CYCLES \mid size, compiler)$	**0.256**	**0.153**	**0.087**

TABLE I
RÉSULTATS DU TEST D'INFORMATION MUTUELLE SUR TROIS BASES D'OBSERVATIONS DIFFÉRENTES. LES NOMBRES EN GRAS DÉNOTENT UNE DÉPENDANCE.

Références

[1] COOK.R.D, WEISBERG.S, An introduction to Regression Graphics, Wiley Series in Probability and Statistics, 1994.

[2] Geiger.D, Heckerman. D, Knowledge representation and inference in similarity networks and Bayesian multinets. Artificial Intelligence, 82, 45–74, 1996.

[3] Lemeire.J *Learning Causal Models of Multivariate Systems and the Value of it for the Performance Modeling of Computer Programs*,Ph.D. thesis,Vrije Universiteit Brussel,2007.

[4] Lemeire.J, Meganck.S, Cartella.F, *Robust Independence-Based Causal Structure Learning in Absence of Adjacency Faithfulness*, Vrije Universiteit Brussel Belgium, Helsinki - Finland, September 2010

[5] Lemeire.J, Dirkx.E, *Causal Performance Models of Computer Systems : Definition and Learning Algorithms*, Vrije Universiteit Brussel, IRIS-TR-0100, 2006, http ://parallel.vub.ac.be.

[6] Pearl.J, *Causality : Models, Reasoning, and Inference*,Cambridge University Press, Cambridge, England,2000.

[7] Pearl.J, *Probabilistic Reasoning in Intelligent Systems : Networks of Plausible Inference.*, Morgan Kaufmann,1988.

[8] Lauritzen, Causal inference from graphical models. Complex Stochastic Systems, pages 63–107, 2001.

[9] Erlend Jensen.R, *Compiler comparison using performance counters, CERN openlab, 2009.*

[10] Spirtes.P, Glymour.C, Scheines.R, *Causation, Prediction, and Search*, The MIT Press,2,2000

[11] Cover.T.M, Thomas.J.A. *Elements of InformationTheory*. John Wiley Sons, Inc., 1991.

[12] Richard.E. Neapolitan,*Learning Bayesian Networks*,Prentice Hall,2003.