Évaluation de différents critères pour la formation de groupes dans le contexte du pair programming distribué : expérimentation dans l'enseignement supérieur

José Colin[1], Sébastien Hoarau[2], and Julien Broisin[1]

[1] IRIT, Université de Toulouse, Toulouse, France
[2] Laboratoire d'Informatique et de Mathématiques, Université de la Réunion, Saint-Denis, France

Résumé. La formation des groupes est une problématique importante de l'apprentissage collaboratif, car des groupes formés de manière appropriée peuvent conduire à de meilleures interactions entre les étudiants et améliorer leur apprentissage. Cette problématique s'applique également à la formation de binômes dans le contexte d'activités d'apprentissage de pair programming. Parmi les méthodes automatiques existantes utilisées pour le pair programming, beaucoup impliquent d'avoir à disposition un certain nombre d'informations personnelles sur les étudiants, ce qui peut complexifier la mise en place de ce genre de dispositif. Sur la base de cette limite, nous proposons une méthode automatique de formation des binômes intégrée à une application de distributed pair programming, et qui s'appuie non seulement sur certaines caractéristiques personnelles des étudiants obtenues avec des données auto-rapportées, mais aussi sur les interactions antérieures des étudiants avec l'application collectées sous forme de traces. Cette méthode a été expérimentée dans le cadre d'un cours d'introduction à la programmation, dans lequel 133 sessions de travail en pair programming ont été réalisées sur notre application de distributed pair programming. Les critères de formation des binômes utilisés par le système pendant cette expérimentation étaient l'expérience en programmation, l'auto-efficacité, le genre et l'engagement. L'expérimentation montre que les binômes homogènes en auto-efficacité ont une compatibilité perçue et une répartition du travail perçue significativement meilleures que les binômes hétérogènes, et que les binômes hétérogènes en experience de programmation ont une performance académique significativement meilleure que les binômes homogènes.

Mots-clés : Apprentissage de l'informatique ; Pair programming distribué ; Formation de groupes ; Conception et évaluation

Abstract. Group formation is an important issue in collaborative learning, as appropriately formed groups can lead to better interactions between students and enhance their learning. This issue also applies to the context of pair programming learning activities. Many of the existing

automatic methods used for pair programming require a certain amount of personal information about the students, which can make setting up this kind of system very complex. Based on these limitations, we present an automatic pair formation method integrated to a distributed pair programming application, using personal characteristics of the students as formation criteria, obtained with self-reported data but also with trace data from the students. We experimented this method as part of an introductory programming course, in which 133 peer programming sessions were carried out on our distributed pair programming application. The criteria for pair formation used by the system during this experiment were programming experience, self-efficacy, gender and engagment. The experiment shows that homogeneous pairs in self-efficacy are significantly better than heterogeneous pairs, both in terms of perceived compatibility and perceived work distribution, and that the heterogeneous pairs in programming experience have a significantly higher academic performance than homogeneous pairs.

Keywords: Computer Science Education ; Distributed Pair Programming ; Group Formation ; Tool Design and Evaluation

1 Introduction

Le pair programming (PP) est une méthode de développement impliquant deux programmeurs : le *Driver*, et le *Navigator*. Le *Driver* travaille activement sur la tâche de programmation et rédige le code, tandis que le *Navigator* observe et apporte son aide au *Driver* en mettant en avant des améliorations possibles. Appliqué à l'enseignement, le PP a de multiples impacts positifs sur les étudiants, comme une plus grande confiance dans la qualité du code, de meilleures compétences en résolution de problèmes [7], ainsi qu'une plus grande satisfaction perçue des étudiants [17]. Dans un contexte où les deux programmeurs sont éloignés géographiquement, le *distributed pair programming* (DPP) permet de réaliser des activités de PP à distance. Le DPP se déroule dans un environnement virtuel où un éditeur de code est partagé entre les deux programmeurs, garantissant que seul le programmeur dans le rôle de *Driver* peut modifier le code source. L'environnement possède également des fonctionnalités pour exécuter du code et permettre au *Driver* et *Navigator* d'inverser leurs rôles. La communication peut se faire de manière textuelle ou vocale.

Pour le PP et le DPP, comme pour l'apprentissage collaboratif de manière générale, la formation des binômes est une problématique importante car elle a un impact significatif sur l'apprentissage et la participation des apprenants [9]. Une solution simple et étudiée dans différents travaux [22,10] consiste à laisser les étudiants choisir leurs binômes : ces études montrent qu'ils ont alors tendance à choisir en fonction de leurs affinités [22]. Cette solution donne un taux de compatibilité perçue important pour les binômes (plus de 95% pour [22] et [10]). Il est également possible que l'enseignant soit en charge de la formation des binômes [12]. Il se base alors sur la connaissance de ses étudiants pour former les

binômes qu'il juge les plus pertinents par rapport aux objectifs d'apprentissage qu'il a fixés.

Cependant, ces solutions ne sont pas toujours applicables. Dans l'enseignement supérieur, les étudiants de première année, souvent en grand nombre, ne se connaissent pas nécessairement. Cela rend difficile la formation des binômes en fonction de l'affinité, et le nombre important d'étudiants rend la tâche complexe et chronophage pour l'enseignant. Dans le cas des MOOCs, les mêmes problèmes se posent. Les étudiants ont encore moins d'opportunités d'apprendre à se connaître, et les enseignants ne sont pas forcément amenés à connaître leurs étudiants. Pourtant, la formation des binômes reste un enjeu majeur pour l'efficacité du PP.

Dans cet article, nous proposons une méthode automatique de formation des binômes dans le cas du DPP. Cette méthode cherche, à partir d'un nombre important d'étudiants, à effectuer des appariements en fonction de leur profil. En particulier, nous étudions leur expérience de programmation, leur sentiment d'auto-efficacité, leur genre, et leur engagement antérieur dans des activités de PP. L'objectif est d'étudier et de comparer l'impact de ces caractéristiques sur la compatibilité perçue par les étudiants avec leur partenaire, la répartition du travail perçue par les étudiants au sein du binôme, ainsi que sur la performance et l'engagement des étudiants dans l'activité de DPP. Notre approche consiste à constituer, pour une même caractéristique du profil des apprenants, des groupes homogènes d'une part et des groupes hétérogènes d'autre part.

Après un état de l'art sur la formation des groupes dans le domaine général de l'apprentissage collaboratif supporté par ordinateur, et du PP en particulier, nous présentons notre système de formation des binômes et les détails de l'algorithme qui le met en œuvre au sein d'une application existante de DPP [3]. L'évaluation de ce système dans le cadre d'une expérimentation menée dans l'enseignement supérieur et les résultats obtenus sont ensuite discutés. Finalement, nous donnons les principales conclusions de ces travaux et les pistes de recherche que nous souhaitons explorer pour améliorer notre système de formation des groupes.

2 État de l'art

Cette section présente dans un premier temps les approches de formation de groupes proposées dans le domaine du Computer Supported Collaborative Learning (CSCL), puis plus spécifiquement dans le cas du pair programming.

2.1 Formation des groupes dans le domaine du CSCL

La formation des groupes dans le domaine du CSCL fait référence aux stratégies, techniques et méthodes mises en oeuvre pour regrouper des individus en fonction de critères afin de créer des groupes qui conduisent les apprenants à de meilleures interactions et qui maximisent l'apprentissage [1]. Dans leur revue de littérature, Borges et al. [1] montrent que sur 106 études portant sur

la formation des groupes en CSCL, les principaux critères utilisés portent sur le profil des étudiants (niveau de connaissance, compétences, genre, etc.) et les préférences des étudiants et/ou des enseignants. Ils montrent également que ces critères sont utilisés pour former les groupes selon deux approches : homogène et hétérogène. L'approche homogène cherche à grouper des étudiants similaires pour un ou plusieurs critères donnés, tandis que l'approche hétérogène cherche à grouper des étudiants différents. L'approche hétérogène est plus étudiée (84 études sur 106) que l'approche homogène (36 études sur 106).

Lorsque le nombre d'étudiants ou le nombre de critères à prendre en compte est élevé, la formation des groupes peut devenir une tâche complexe à réaliser sans support informatique. Pour pallier ce problème, on peut, à la place d'une approche manuelle où l'enseignant et/ou les étudiants sont en charge de former les groupes, passer à une approche automatique. Pour cela, De nombreuses études proposent différentes méthodes algorithmiques [4]. Les algorithmes utilisant des modèles probabilistes sont les plus utilisés. Ils permettent notamment de gérer un nombre important de variables.

2.2 Études existantes sur la formation des binômes en pair programming

Dans le cas du pair programming, comme dans celui du CSCL, les études existantes sur la formation des binômes utilisent soit des méthodes manuelles en laissant les étudiants ou les enseignants choisir les binômes [22,12], soit des méthodes automatiques [2,5]. Les critères de formation des binômes portant sur le niveau de connaissance et de compétence des étudiants sont largement utilisés [20,5,6,21], ainsi que les approches homogène et hétérogène. Par ailleurs, l'auto-efficacité est également utilisée comme critère de formation des binômes dans l'étude proposée par Tsai et al. [19].

Plusieurs études [8,12,21,2] utilisent les traits de personnalité des étudiants comme critère pour la formation des binômes en s'appuyant sur le modèle Myers-Briggs Type Indicator (MBTI) [14]. Choi et al. [2] ont montré que des binômes diversifiés en termes de personnalité font preuve d'une plus grande productivité que les binômes similaires ou opposés. Cependant, la mesure des traits de personnalité nécessite de faire passer de longs questionnaires tels que le questionnaire MBTI, ce qui n'est pas applicable à tous les contextes.

D'autre part, pour évaluer l'efficacité d'une méthode de formation de binômes, plusieurs indicateurs sont utilisés dans la littérature. La compatibilité perçue entre les deux membres du binôme est mesurée via des questionnaires post-expérimentation dans plusieurs études [12,6,5,21]. La performance des binômes dans les tâches de PP est évaluée en mesurant le temps nécessaire au binôme pour réaliser la tâche de programmation [8] et/ou en évaluant directement le code produit par le binôme [5,8].

Parmi les recherches existantes, nous n'avons pas trouvé d'études exploitant les traces d'interaction ou les learning analytics pour la formation des binômes. Par exemple, l'utilisation de l'engagement pour la formation des binômes, mesuré à partir des interactions de l'utilisateur, n'a pas, à notre connaissance, été étudiée

dans la littérature. Exploiter les traces d'interaction peut permettre d'obtenir des mesures plus objectives sur le comportement réel des étudiants en PP qu'avec des données auto-rapportées.

Dans la suite de cet article, nous essayons donc d'apporter des éléments de réponse à la question de recherche (QR) suivante : Comment la formation automatique de groupes homogènes et hétérogènes en termes d'auto-efficacité des apprenants, d'expérience en programmation, d'engagement antérieur dans des activités de pair programming, et de genre, affecte la compatibilité et la répartition du travail perçues, l'engagement, et la performance des apprenants dans le DPP ?

3 Système de formation des binômes

Cette section introduit d'abord notre application de distributed pair programming existante, puis expose le nouveau système de formation des binômes intégré dans cette application.

3.1 L'application de distributed pair programming

Le système de formation automatique des binômes que nous proposons est intégré directement dans notre application de distributed pair programming [3]. C'est une application web permettant de réaliser des travaux pratiques en Python. Au niveau de l'interface visible par les étudiants, toutes les fonctionnalités importantes pour la mise en œuvre du DPP sont implémentées : un éditeur de code partagé, un système de rôle pour différencier le *Driver* et le *Navigator* et permettre les changements de rôle, une console Python pour l'exécution du code, et une messagerie textuelle instantanée. Pour l'enseignant, l'application met à disposition une interface pour créer et modifier des activités pédagogiques. Un certain nombre de traces d'interactions sont collectées : connexion et déconnexion des utilisateurs ; demandes de changement de rôle et les réponses associées ; captures du code source Python produit ; interactions avec la console Python ; messages échangés dans le canal textuel. Elles sont horodatées et formatées en format JSON.

L'application a déjà été utilisée dans le cadre d'expérimentations visant à évaluer son utilisabilité [3]. Elle est accessible avec le protocole LTI [13], ce qui permet aux étudiants d'y accéder depuis une plate-forme d'apprentissage telle que Moodle.

3.2 Fonctionnement général du système de formation des binômes

Pour démarrer une session de travail en PP en utilisant le système de formation des binômes, l'enseignant commence par créer une activité sur l'application, en renseignant une consigne pour un exercice de programmation. Les étudiants se connectent ensuite simultanément sur cette activité. Ils sont alors placés dans

une file d'attente, qui se vide à mesure que les binômes sont formés et redirigés vers l'environnement de DPP.

Le système dispose des informations nécessaires sur le profil des étudiants pour effectuer les calculs de score d'appariement. À intervalle régulier, le système va prendre l'ensemble des combinaisons d'étudiants possibles dans la file d'attente et calculer pour chacune un score d'appariement selon une méthode de calcul du score d'appariement choisie. La combinaison ayant le plus grand score est alors sélectionnée pour former un binôme. Les 2 étudiants concernés quittent la file d'attente, puis sont redirigés vers l'activité de DPP. Cette opération est répétée jusqu'à ce que la file d'attente soit vide (ou jusqu'à ce qu'elle ne contienne plus qu'un seul étudiant). L'implémentation de ce système est décrite dans l'algorithme 1.

Algorithme 1 Algorithme de formation des binômes

$etudiants \leftarrow fileAttente$ ▷ L'ensemble des étudiants dans la file d'attente
while $size(etudiants) \geq 2$ **do**
  $maxScore \leftarrow 0$
  $meilleureCombinaison \leftarrow \emptyset$
  for $(etudiant1, etudiant2) \in etudiants \times etudiants$ **do**
    if $etudiant1 \neq etudiant2$ **then**
      $scoreAppariement \leftarrow calculScoreAppariement(etudiant1, etudiant2)$
      if $scoreAppariement > maxScore$ **then**
        $maxScore \leftarrow scoreAppariement$
        $meilleureCombinaison \leftarrow \{etudiant1, etudiant2\}$
  $formerBinome(meilleureCombinaison)$
  $etudiants \leftarrow etudiants \setminus meilleureCombinaison$

Il est également possible d'introduire dans ce système un seuil pour le score d'appariement. Dans ce cas-là, si parmi toutes les combinaisons d'étudiants, aucune ne dépasse ce seuil, alors aucun binôme n'est formé, et le système attend pendant une durée prédéfinie avant de recommencer à chercher des binômes. Cette attente permet d'éviter de former des binômes trop rapidement et de laisser l'opportunité à la file d'attente de se remplir afin de trouver de meilleures combinaisons d'étudiants. Si, après plusieurs attentes consécutives, aucune combinaison ne dépasse toujours pas le seuil, il est alors progressivement diminué jusqu'à ce que la situation se débloque. Cet ajout est pertinent dans le cas où les étudiants ne se connectent pas forcément tous en même temps à l'application. Pour l'expérimentation présentée dans cette étude, nous n'avons pas utilisé de système de seuil.

3.3 Calcul du score d'appariement

Le calcul du score d'appariement se fait selon une méthode choisie. Une méthode est associée à un critère de formation et elle a un objectif : maximiser l'homogénéité entre les deux étudiants sur ce critère ou maximiser l'hétérogénéité sur ce

critère. Pour une méthode qui maximise l'homogénéité, le score d'appariement s'obtient avec la formule suivante :

$$scoreAppariement = \frac{1}{\max(0.01, \mid critereEtudiant1 - critereEtudiant2 \mid)}$$

Pour une méthode qui maximise l'hétérogénéité, le score d'appariement s'obtient avec la formule suivante :

$$scoreAppariement = \mid critereEtudiant1 - critereEtudiant2 \mid$$

4 Méthode de conduite de la recherche

4.1 Configuration expérimentale

L'étude que nous avons menée a été réalisée à travers notre application de DPP [3] décrite dans la section 3.1, dans le cadre d'un cours d'introduction au langage Python à l'Université de la Réunion. Cette expérimentation a impliqué une cohorte d'étudiants inscrits en première année de licence informatique, et une cohorte d'étudiants inscrits en première année de licence de mathématiques. Chaque cohorte a utilisé notre plate-forme pendant deux séances de cours magistraux dispensées en présentiel par le même enseignant. Le tableau 1 donne certaines données démographiques sur les étudiants ayant participé à cette expérimentation.

Table 1. Données démographiques sur les participants

Nombre total	Nombre de L1 Informatique	Nombre de L1 Mathématiques	Nombre de filles	Nombre de garçons	Moyenne d'âge
102	82	20	8	94	19,6

Durant la première séance de cours magistral, l'enseignant a tout d'abord présenté notre plate-forme afin d'exposer ses principales fonctionnalités. Ensuite, les étudiants ont disposé de 20 minutes pour réaliser une activité de PP conçue par l'enseignant. Durant la deuxième séance de cours magistral, les étudiants ont participé à deux activités de PP, chacune d'une durée de 40 minutes.

Chaque activité de PP était centrée sur des notions vues précédemment dans le cadre du cours d'introduction au langage Python. Voici un exemple de consigne pour une activité de PP en Python : "Ecrire une fonction ‘echange‘, qui intervertit deux cases d'un tableau donné. ‘nombres = [45, 34, 12, 23, 57]‘ echange(nombres,2,4)‘ doit modifier nombres en : ‘[45, 34, 57, 23, 12]‘" Elles étaient accessibles depuis la plate-forme d'apprentissage Moodle de l'Université Anonyme. Ainsi, les étudiants étaient automatiquement connectés sur notre plate-forme via le protocole LTI (voir section 3.1). Lors de leur première connexion

à la plate-forme (i.e., lors de la réalisation de la première activité), les étudiants devaient remplir le questionnaire pré-expérimental Q1 concernant leur expérience en programmation et leur sentiment d'auto-efficacité. Après avoir réalisé chacune des trois activités, ils devaient remplir le questionnaire post-expérimentation Q2 concernant la répartition du travail perçue au sein de leur binôme ainsi que la compatibilité perçue avec leur partenaire.

Au début de chaque activité, les binômes ont été formés par le système de formation automatique décrit dans la section 3.2. Le système s'est appuyé sur différents critères de formation (i.e., l'expérience en programmation des apprenants ; leur sentiment d'auto-efficacité ; leur genre ; leur engagement antérieur dans des activités de PP), en essayant d'équilibrer autant que possible le nombre de binômes formés selon chaque critère de formation, en fonction des données disponibles sur les étudiants. Les étudiants se sont donc vus attribuer un binôme localisé à n'importe quel endroit de l'amphithéâtre, et devaient communiquer via le canal textuel intégré à l'application. Pendant la première séance, les critères de formation utilisés étaient l'expérience en programmation, l'auto-efficacité et le genre. Pour la deuxième séance, les critères utilisés étaient l'engagement antérieur, l'auto-efficacité et le genre.

4.2 Données collectées

Les données collectées sont anonymisées. Seul l'identifiant Moodle transmis par LTI est collecté, ne permettant pas de révéler l'identité des étudiants.

Données auto-rapportées. Le questionnaire pré-expérimental Q1, dispensé via notre plate-forme DPP lors de la première connexion seulement et disponible sur recherche.data.gouv.fr., a pour objectif de mesurer trois critères utilisés dans le système de formation des binômes : l'expérience en programmation des apprenants, leur auto-efficacité et leur genre. L'expérience en programmation est obtenue à partir du questionnaire proposé par Siegmund et al. [18], traduit en français. Ce questionnaire comprend 15 questions au total, parmi lesquelles 8 questions à choix uniques et 7 questions de Likert à 5 niveaux. Le **score d'expérience de programmation** est obtenu en calculant la moyenne des réponses à ces questions. Nous définissons, pour chaque question à choix unique, une méthode pour obtenir un score. Le détail du calcul est disponible sur recherche.data.gouv.fr. Nous avons utilisé le questionnaire proposé par Tsai et al. [19], traduit en français, pour mesurer l'auto-efficacité des apprenants. Ce questionnaire est constitué de 16 questions de Likert à 6 niveaux. Le **score d'auto-efficacité** est obtenu en calculant la moyenne des réponses formulées par le répondant. Enfin, **le genre** est obtenu à travers 1 question à choix unique.

Le questionnaire post-expérimentation Q2, également dispensé via notre plate-forme DPP mais cette fois à la fin de chaque activité de PP, vise à mesurer leur perception de la répartition du travail avec leur partenaire, ainsi que la compatibilité perçue par les étudiants au sein de leur binôme . Ce questionnaire apparaît en Annexe A.. Nous nous sommes appuyés sur le questionnaire de Williams et

al. [21], traduit en français, qui vise à mesurer lors de situations de PP, (i) la répartition du travail perçue par les étudiants à l'aide de 3 questions de Likert à 5 niveaux, et (ii) la compatibilité perçue à l'aide d'1 question de Likert à 3 niveaux. Le **score de répartition du travail perçue *au sein du binôme*** est calculé en faisant la moyenne des réponses fournies par les 2 membres du binôme aux 3 questions R1, R2 et R3. Le **score de compatibilité perçue *au sein du binôme*** est calculé en suivant la même démarche, en calculant la moyenne de la réponse des 2 membres à la question C1.

Données quantitatives. Comme mentionné dans la section 3.1, notre application de DPP collecte différentes traces d'interactions. Ces données sont utilisées pour calculer 2 indicateurs : la performance académique du binôme et son engagement dans l'activité de PP. La **performance académique** correspond à un score obtenu à partir du code source final produit par le binôme par rapport à l'activité proposée par l'enseignant. Elle correspond à l'évaluation manuelle du code source selon un barème établi par l'enseignant ayant proposé l'activité. Afin d'éviter les biais de notation, un seul et unique correcteur a procédé à l'évaluation de l'ensemble des codes sources. Ainsi, pour chacune des 3 activités, chaque binôme a obtenu un score de performance académique.

L'engagement du binôme dans l'activité de PP est obtenu en combinant différentes variables calculées pour chaque membre du binôme :

- Le nombre de caractères insérés/supprimés (*nbChar*). Cette valeur est obtenue en comparant les différences successives entre chaque capture du contenu de l'éditeur de code. Pour chaque modification du code, la différence en nombre de caractères avant et après la modification est ajoutée au total de l'étudiant auteur de la modification. Cette différence est obtenue avec l'algorithme de Myers [15]. Cette façon de faire permet de bien distinguer la contribution des deux membres du binôme.
- Le nombre d'exécutions du code et de commandes tapées dans la console Python (*nbCommand*).
- Le nombre de demandes de changement de rôle, et de réponses à une demande de changement de rôle (*nbRoleChange*).
- Le nombre de messages écrits dans le canal de communication textuel (*nbMessage*).

Ces variables sont ensuite normalisées en divisant par les valeurs maximales obtenues dans le jeu de données lors de la première séance de l'expérimentation ($nbCharMax = 920$, $nbCommandMax = 19$, $nbRoleChangeMax = 5$, $nbMessageMax = 31$). Ainsi, pour chaque étudiant, nous calculons un score d'engagement compris entre 0 et 1 en appliquant la formule suivante :

$$engagement = \frac{(\frac{nbChar}{920} + \frac{nbCommand}{19} + \frac{nbRoleChange}{5} + \frac{nbMessage}{31})}{4}$$

Finalement, le **score d'engagement du binôme** dans l'activité de PP est obtenu en calculant la moyenne de l'engagement individuel des deux membres du

binôme. Notons ici que le score d'engagement individuel de la formule ci-dessus a également été utilisé, lors de la deuxième séance, pour le critère de formation d'engagement antérieur.

4.3 Analyse des données

Pour répondre à notre question de recherche concernant l'impact de binômes homogènes et hétérogènes en termes d'auto-efficacité, d'expérience en programmation, de genre et d'engagement antérieur dans des activités de PP sur la compatibilité et la répartition du travail perçues, l'engagement et la performance des apprenants, nous avons rapporté les moyennes, déviations standard et médianes. Nous avons ensuite cherché à identifier si des différences significatives existent entre le groupe homogène et le groupe hétérogène sur chacun des critères de formation utilisés. Nous avons utilisé des tests U de Mann-Whitney car nos données n'étaient pas normalement distribuées. Ces tests ont été réalisés pour les 4 variables *perception de la répartition du travail*, *perception de la compatibilité*, *performance académique*, et *engagement dans l'activité de pair programming*.

Pour classer les binômes dans les groupes homogène et hétérogène dans notre analyse, nous avons placé, pour chaque critère de formation, la moitié des binômes ayant la différence la plus faible sur le critère dans le groupe hétérogène, et l'autre moitié dans le groupe homogène. Pour le critère du genre, seuls 8 binômes hétérogènes (homme/femme) ont pu être formés, les deux groupes ne sont donc pas de même taille.

5 Résultats

Au total, 133 sessions de travail de DPP, avec des binômes différents à chaque fois, ont été prises en compte dans notre étude. 51 et 82 sessions ont respectivement été réalisées pendant la première et la deuxième séance de cours magistral. Le tableau 2 donne le nombre de binômes créés lors de l'expérimentation pour chaque critère de formation.

Table 2. Nombre de binômes par critère de formation

	Expérience en programmation	Auto-efficacité	Genre	Engagement antérieur
Nombre de binômes	21	50	35	27

Le nombre de binômes formés n'est pas le même pour chacun des critères. En effet, le critère "expérience en programmation" n'a été utilisé que pendant la première séance de cours magistral, où un nombre moins important de sessions de travail ont été réalisées. Aussi, le critère "engagement antérieur" n'a été utilisé que lors de la deuxième séance de cours magistral. En effet, ce critère ne peut être

calculé automatiquement que lorsqu'un étudiant a réalisé au moins une activité de PP sur notre plate-forme (voir section 4.3).

Les figures 1, 2, 3 et 4 présentent les moyennes obtenues pour les indicateurs de compatibilité perçue, de performance académique et d'engagement en fonction des différentes méthodes de formation des groupes utilisées. La figure 1 ne fait pas apparaître de différences importantes dans les moyennes de compatibilité perçue entre les groupes homogènes et hétérogènes en expérience de programmation et en genre, mais fait apparaître des différences plus importantes pour l'auto-efficacité favorisant le groupe homogène et pour l'engagement favorisant le groupe hétérogène. La figure 2 montre de légères différences pour le genre et l'engagement, et une différence un peu plus importante pour l'auto-efficacité dans la répartition du travail perçue moyenne. Comme pour la compatibilité perçue, c'est le groupe homogène qui obtient de meilleurs résultats pour l'auto-efficacité. La figure 3 montre une différence significative pour l'expérience de programmation en faveur du groupe hétérogène dans la performance académique du binôme, mais pas de différence pour l'auto-efficacité, le genre et l'engagement. Enfin, la figure 4 met en évidence des différences importantes pour l'expérience en programmation et l'engagement, et pas de différences pour l'auto-efficacité et le genre dans l'engagement du binôme.

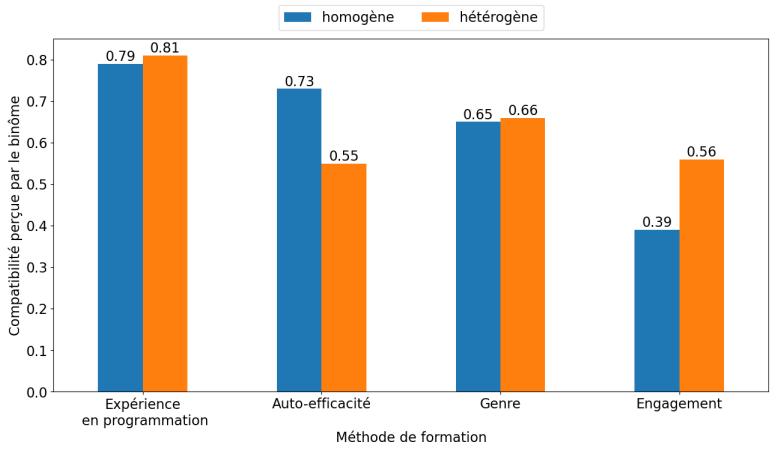

Figure 1. Compatibilité perçue moyenne des binômes en fonction des méthodes de formations utilisées

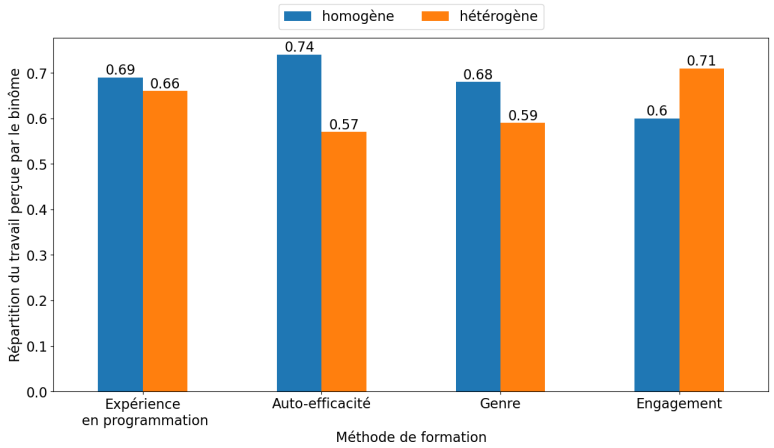

Figure 2. Répartition du travail perçue moyenne des binômes en fonction des méthodes de formations utilisées

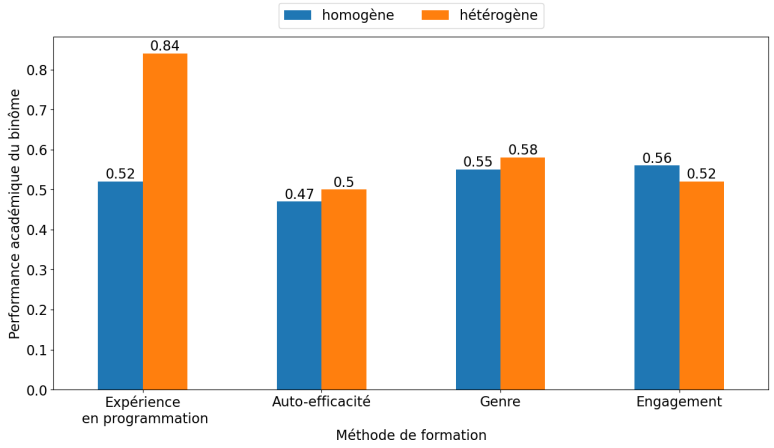

Figure 3. Performance académique moyenne des binômes en fonction des méthodes de formations utilisées

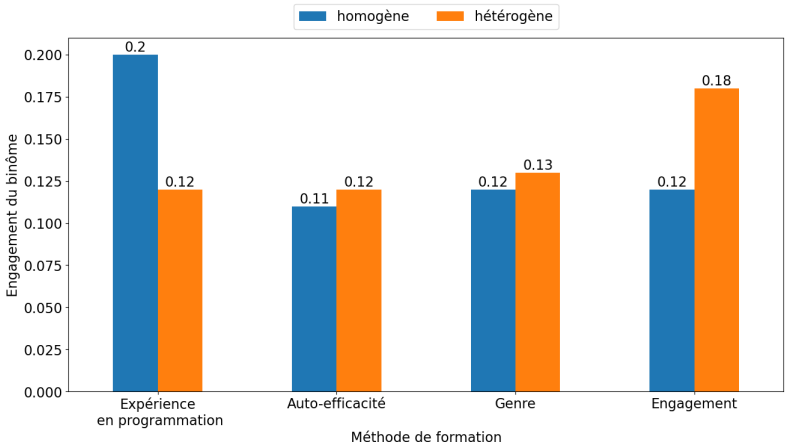

Figure 4. Engagement moyen des binômes en fonction des méthodes de formations utilisées

La table 3 présente les résultats des tests U de Mann-Whitney comparant les groupes homogènes et les groupes hétérogènes pour chaque critère de formation, ainsi que le détail des moyennes, déviations standard et médianes pour les indicateurs de compatibilité et de répartition du travail perçus, d'engagement et de performance des apprenants. Cette table fait apparaître 3 résultats montrant une différence significative ($p < .05$) entre les groupes homogènes et hétérogènes : les binômes ayant une auto-efficacité homogène ont une compatibilité perçue et une répartition du travail perçue significativement plus importante que les binômes hétérogènes, et les binômes ayant une expérience en programmation hétérogène ont une performance académique significativement meilleure que les binômes homogènes.

6 Discussion

Nous observons que les binômes avec une auto-efficacité homogène ont une compatibilité perçue significativement plus importante que les binômes avec une auto-efficacité hétérogène ($p = 0.01$). On obtient un résultat similaire pour la répartition du travail perçue ($p = 0.01$). Ce résultat va dans le sens d'une étude menée par Katira et al., dans laquelle les auteurs ont montré que les binômes ayant une compétence technique perçue homogène étaient plus compatibles que les binômes hétérogènes [12]. Ces résultats suggèrent que ce critère de formation peut être utilisé pour maximiser la répartition du travail au sein des binômes.

On peut également faire le rapprochement entre ce résultat et des études montrant une meilleure compatibilité perçue pour les binômes homogènes en compétence de programmation [5,11,20]. Cependant, notre étude ne montre pas

Table 3. Résultats des tests U de Mann-Whitney comparant les groupes homogènes et les groupes hétérogènes pour chaque critère de formation

	Compatibilité perçue par le binôme					Répartition du travail perçue par le binôme					Performance académique du binôme					Engagement du binôme				
Méthode de formation	M	SD	Mdn	U	p	M	SD	Mdn	U	p	M	SD	Mdn	U	p	M	SD	Mdn	U	p
Expérience en programmation homogène	0.79	0.25	1	210	0.77	0.69	0.66	1	248.5	0.47	0.52	0.41	0.50	127.5	**0.01***	0.203	0.205	0.106	250	0.46
Expérience en programmation hétérogène	0.81	0.25	1			0.66	0.36	0.83			0.84	0.30	1			0.120	0.085	0.132		
Auto-efficacité homogène	0.73	0.31	1	1604	**0.01***	0.74	0.32	0.83	1634	**0.01***	0.47	0.42	0.33	1210	0.78	0.111	0.117	0.082	1148.5	0.48
Auto-efficacité hétérogène	0.55	0.34	0.5			0.57	0.34	0.58			0.50	0.42	0.5			0.123	0.117	0.113		
Genre homogène	0.65	0.29	0.5	424.5	0.91	0.68	0.34	0.75	490	0.41	0.55	0.42	0.67	436	0.96	0.118	0.087	0.104	387	0.53
Genre hétérogène	0.66	0.30	0.5			0.59	0.37	0.71			0.58	0.31	0.58			0.125	0.087	0.115		
Engagement homogène	0.39	0.29	0.5	282	0.13	0.60	0.39	0.75	319.5	0.43	0.56	0.36	0.5	389.5	0.66	0.123	0.110	0.110	280	0.15
Engagement hétérogène	0.56	0.45	0.5			0.71	0.31	0.67			0.52	0.34	0.5			0.182	0.145	0.179		

*$p < .05$

de différence significative sur la compatibilité perçue pour le critère d'expérience en programmation. Une limite conduisant possiblement à ce résultat est discutée dans la section 7.

Pour l'engagement, malgré des écarts de moyenne importants entre le groupe homogène et hétérogène sur deux des critères de formation, les résultats des tests T indépendants ne montrent pas de différences significatives. Cela peut s'expliquer par la déviation standard importante que l'on observe sur cet indicateur.

En ce qui concerne la performance académique, les binômes hétérogènes en expérience de programmation ont obtenu des résultats significativement supérieurs aux binômes homogènes ($p = 0{,}01$). Ce résultat est cohérent avec des études suggérant qu'associer des individus ayant des expériences diverses peut conduire à de bons résultats. [16].

7 Limitations

Dans cette étude, nous avons cherché à reproduire les conditions du DPP, en éloignant physiquement les étudiants d'un même binôme. Même si les étudiants n'étaient pas côte à côte, le fait qu'ils soient présents dans la même salle de cours a pu en amener certains à vouloir communiquer directement avec leurs binômes, ce qui a pu perturber le déroulement des sessions de travail.

L'expérience en programmation a été mesurée à l'aide de réponses à un questionnaire portant sur le profil des étudiants. Or, les étudiants ayant pour beaucoup le même âge et une expérience en programmation similaire venant essentiellement de leur expérience au lycée, la déviation standard sur cet indicateur est très faible. Les groupes considérés comme homogènes ou hétérogènes sur ce critère dans notre étude ne le sont en réalité que très peu. Il aurait été préférable d'utiliser une cohorte d'étudiants plus diverse.

Le calcul de l'indicateur d'engagement a été réalisé de façon arbitraire, notamment au niveau du poids accordé à chaque variable. Il pourrait être intéressant de proposer une autre formule pour le calcul de l'indicateur d'engagement en mettant en œuvre une démarche scientifique pour déterminer les poids adéquats pour chaque variable.

Pour les binômes hétérogènes en genre, la petite taille de l'échantillon (8 binômes) rend nos résultats difficilement généralisables.

8 Conclusion

Dans cet article, nous proposons un système de formation des binômes, explorant plusieurs méthodes de formation cherchant à maximiser l'homogénéité ou l'hétérogénéité sur des caractéristiques personnelles des étudiants. Nous avons utilisé ce système dans une expérimentation où 133 sessions de travail en distributed pair programming ont eu lieu. Parmi les méthodes de formation utilisées dans cette étude, aucune ne semble être meilleure sur la totalité de nos indicateurs (compatibilité perçue, répartition du travail perçue, performance académique et

engagement). Cependant, les binômes homogènes en auto-efficacité ont obtenu des résultats significativement meilleurs sur deux indicateurs : la compatibilité et la répartition du travail perçue. Toutes les données de cette expérimentation sont disponibles en ligne sur recherche.data.gouv.fr.

Cette étude n'utilise que des méthodes utilisant un unique critère de formation. Il pourrait être intéressant, en réutilisant les mêmes critères pour la formation des binômes, d'utiliser des approches hybrides [1], utilisant plusieurs critères simultanément.

Références

1. Borges, S., Mizoguchi, R., Bittencourt, I.I., Isotani, S. : Group formation in cscl : A review of the state of the art. In : Cristea, A.I., Bittencourt, I.I., Lima, F. (eds.) Higher Education for All. From Challenges to Novel Technology-Enhanced Solutions. pp. 71–88. Springer International Publishing, Cham (2018)
2. Choi, K.S., Deek, F.P., Im, I. : Exploring the underlying aspects of pair programming : The impact of personality. Information and Software Technology **50**(11), 1114–1126 (2008). https://doi.org/https://doi.org/10.1016/j.infsof.2007.11.002, https://www.sciencedirect.com/science/article/pii/S0950584907001292
3. Colin, J., Hoarau, S., Declercq, C., Broisin, J. : Design and evaluation of a web-based distributed pair programming tool for novice programmers. In : Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1. p. 527–533. ITiCSE 2024, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3649217.3653571, https://doi.org/10.1145/3649217.3653571
4. Cruz, W.M., Isotani, S. : Group formation algorithms in collaborative learning contexts : A systematic mapping of the literature. In : Baloian, N., Burstein, F., Ogata, H., Santoro, F., Zurita, G. (eds.) Collaboration and Technology. pp. 199–214. Springer International Publishing, Cham (2014)
5. Ömer Demir, Seferoglu, S.S. : The effect of determining pair programming groups according to various individual difference variables on group compatibility, flow, and coding performance. Journal of Educational Computing Research **59**(1), 41–70 (2021). https://doi.org/10.1177/0735633120949787, https://doi.org/10.1177/0735633120949787
6. Dou, W., He, W. : Compatibility and requirements analysis of distributed pair programming. In : 2010 Second International Workshop on Education Technology and Computer Science. vol. 1, pp. 467–470 (2010). https://doi.org/10.1109/ETCS.2010.367
7. Faja, S. : Pair programming as a team based learning activity : A review of research. Issues in Information Systems **XII**, 207–216 (01 2011)
8. Hannay, J.E., Arisholm, E., Engvik, H., Sjoberg, D.I. : Effects of personality on pair programming. IEEE Transactions on Software Engineering **36**(1), 61–80 (2010). https://doi.org/10.1109/TSE.2009.41
9. Isotani, S., Inaba, A., Ikeda, M., Mizoguchi, R. : An ontology engineering approach to the realization of theory-driven group formation. International Journal of Computer-Supported Collaborative Learning **4**, 445–478 (2009)

10. Jacobson, N., Schaefer, S.K. : Pair programming in cs1 : overcoming objections to its adoption. SIGCSE Bull. **40**(2), 93–96 (Jun 2008). https://doi.org/10.1145/1383602.1383643, https://doi.org/10.1145/1383602.1383643
11. Jensen, R. : A pair programming experience. The Journal of Defense Software Engineering **16**(3), 22–24 (2003), http://www.stsc.hill.af.mil/crosstalk/2003/03/jensen.html
12. Katira, N., Williams, L., Wiebe, E., Miller, C., Balik, S., Gehringer, E. : On understanding compatibility of student pair programmers. SIGCSE Bull. **36**(1), 7–11 (Mar 2004). https://doi.org/10.1145/1028174.971307, https://doi.org/10.1145/1028174.971307
13. Lti. https://www.1edtech.org/standards/lti (2024)
14. Meyers, B.I., Meyers, P.B. : Gifts differing understanding personality type (1995)
15. Myers, E.W. : Ano(nd) difference algorithm and its variations. Algorithmica **1**(1), 251–266 (Nov 1986). https://doi.org/10.1007/BF01840446, https://doi.org/10.1007/BF01840446
16. Salinas, E.Y., Williams, A.E., King, C.E. : Effect of controlling group heterogeneity on student performance in a graphical programming course. In : 2019 IEEE Frontiers in Education Conference (FIE). pp. 1–8 (2019). https://doi.org/10.1109/FIE43999.2019.9028593
17. Salleh, N., Mendes, E., Grundy, J. : Empirical studies of pair programming for cs/se teaching in higher education : A systematic literature review. IEEE Transactions on Software Engineering **37**(4), 509–525 (2011)
18. Siegmund, J., Kästner, C., Liebig, J., Apel, S., Hanenberg, S. : Measuring and modeling programming experience. Empirical Software Engineering **19**(5), 1299–1334 (Oct 2014). https://doi.org/10.1007/s10664-013-9286-4, https://doi.org/10.1007/s10664-013-9286-4
19. Tsai, M.J., Wang, C.Y., Hsu, P.F. : Developing the computer programming self-efficacy scale for computer literacy education. Journal of Educational Computing Research **56**(8), 1345–1360 (2019). https://doi.org/10.1177/0735633117746747, https://doi.org/10.1177/0735633117746747
20. Van Toll III, T., Lee, R., Ahlswede, T. : Evaluating the usefulness of pair programming in a classroom setting. In : 6th IEEE/ACIS International Conference on Computer and Information Science (ICIS 2007). pp. 302–308 (2007). https://doi.org/10.1109/ICIS.2007.96
21. Williams, L., Layman, L., Osborne, J., Katira, N. : Examining the compatibility of student pair programmers. In : AGILE 2006 (AGILE'06). pp. 10 pp.–420 (2006). https://doi.org/10.1109/AGILE.2006.25
22. Xinogalos, S., Satratzemi, M., Chatzigeorgiou, A., Tsompanoudi, D. : Student perceptions on the benefits and shortcomings of distributed pair programming assignments. In : 2017 IEEE Global Engineering Education Conference (EDUCON). pp. 1513–1521 (2017). https://doi.org/10.1109/EDUCON.2017.7943050

Annexe A. Questionnaire Q2

- R1. Votre partenaire de pair programming a-t-il fait sa juste part du travail ?
- R2. Votre partenaire a-t-il suivi le modèle de pair programming (en alternant les rôles de driver et de navigator) ?
- R3. Votre partenaire de pair programming a-t-il contribué à la réalisation de l'exercice ? A-t-il coopéré ?
- C1. D'après vous, à quel point vous et votre partenaire étaient compatibles ?