Fonds de recherche
Société et culture
Québec

14 juillet 2024

Rapport de recherche

PROGRAMME ACTIONS CONCERTÉES

Intégration de la programmation en mathématiques et en sciences au secondaire: quels enjeux pour la formation des enseignants.

Chercheure principale
Fabienne Venant, UQÀM

Cochercheur
Raoul Kamga Kouamkam, UQÀM

Collaboratrices ou collaborateurs
Martin Baril, CSS de la Capitale
Sonya Fiset, RECIT_MST

Établissement gestionnaire de la subvention
Université du Québec À Montréal

Numéro du projet de recherche
293415

Titre de l'Action concertée
Programme de recherche-action sur le numérique en éducation et en enseignement supérieur

Partenaire(s) de l'Action concertée
Le ministère de l'Éducation du Québec
Et le Fonds de recherche du Québec – Société et culture (FRQSC)

Partie A – Contexte de la recherche

Problématique

Beaucoup d'enjeux citoyens actuels reposent sur la capacité du système éducatif de permettre à tous d'acquérir des compétences de calibre mondial en littératie, en numératie et en sciences, tout en développant la conscience citoyenne et éthique. Les avancées informatiques récentes, notamment en termes de stockage et d'analyse de données numériques ont fait naître le besoin de démocratiser des compétences informatiques avancées, comme la programmation et la robotique. Parmi la maîtrise des compétences numériques attendues de la part de la main-d'œuvre ou des citoyens, celles liées à l'usage de la programmation informatique sont de plus en plus au cœur des débats éducatifs (Barma, 2018). Cependant, l'intégration de l'informatique dans les milieux scolaires constitue encore un défi. Pourtant reconnue comme une compétence du 21e siècle (Romero et al., 2017), la programmation, et la pensée informatique qu'elle permet de développer (Wing, 2011; Wing, 2008) peinent à prendre leur place au sein des programmes de formation. La pensée informatique constitue pourtant un outil particulièrement intéressant pour faire le lien entre les compétences disciplinaires, les compétences numériques et l'éducation citoyenne. (Henda, 2017; Karsenti & Bugmann, 2017). Les avancées technologiques numériques ont à la fois rendu possible et complexifié son intégration à l'école, en amenant sur le marché éducatif numérique des outils plus conviviaux mais de plus en plus nombreux (Scratch, Blockly, SageMath, Thymio, Ozobot, Lego EV3…). Les avenues sont innombrables et l'enseignant peut rapidement se sentir perdu face au foisonnement des outils. L'intégration de la pensée informatique passe par l'accompagnement des enseignants dans l'appropriation des outils, de la programmation informatique, et des liens qu'ils permettent de faire avec les approches d'enseignement ouvertes comme la résolution de problèmes individuelle ou collaborative (Barma, 2018).

C'est pourquoi nous avons réalisé une recherche visant à développer et expérimenter de nouveaux modes d'action auprès des enseignants pour faciliter l'intégration de la programmation informatique en classe de

mathématiques. Ce projet répond aux objectifs généraux de l'appel de propositions visant le développement des connaissances sur les usages du numérique les plus susceptibles de favoriser la réussite éducative de tous et de toutes. Notre recherche repose sur la collaboration avec des enseignants au primaire, au secondaire et à l'éducation aux adultes. Elle correspond en cela à la visée multi-ordre et multi-âges de l'appel à propositions. Elle répond plus particulièrement au besoin d'accompagnement et de formation pour le personnel des réseaux de l'éducation et de l'enseignement supérieur exprimé dans l'axe 6. Dans le contexte actuel de la pénurie d'enseignants, il est de plus en plus difficile de libérer les enseignants pour des journées de perfectionnement. Il est donc primordial de proposer de nouveaux modes d'accompagnement pour les enseignants qui souhaitent s'engager dans le virage numérique.

Objectifs poursuivis

Notre premier objectif était d'étudier les possibilités offertes par un partenariat entre le monde universitaire et le milieu professionnel. Nous avons pour cela travaillé de façon étroite avec le RECIT-MST, réseau dont les mandats sont en lien avec l'intégration du numérique dans l'apprentissage des élèves du primaire et du secondaire dans le cadre du programme de formation de l'école québécoise, et le Centre de Services Scolaire de la Capitale. Ensemble, nous avons cherché à mettre en place des formations s'appuyant sur des données de recherche pour promouvoir les pratiques pertinentes pour l'intégration de la programmation informatique et de la robotique pédagogique dans l'enseignement secondaire. Notre recherche vise à relever trois défis : la création de ressources, la formation des enseignants et l'étude des pratiques liées à l'intégration de la programmation informatique dans l'enseignement des mathématiques. Elle poursuit plus particulièrement trois objectifs précis :

1. Recenser les travaux existants (incluant les travaux précurseurs des années 80) afin de documenter et d'analyser les ressources et usages prescrits pour l'intégration de la programmation informatique et de la robotique pédagogique au secondaire.

2. Développer des tâches de programmation informatique pertinentes pour l'enseignement des mathématiques.
3. Développer et expérimenter de nouveaux modes d'action auprès des enseignants pour faciliter l'intégration de la programmation informatique et de la robotique.

Un quatrième objectif figurait dans le devis initial, il était rédigé comme suit :

> Dégager les critères d'efficacité et de pertinence des pratiques et des ressources en programmation informatique et robotique pédagogique au secondaire et construire des ressources qui mettent en œuvre ces critères.

Nous n'avons pas pu réaliser cet objectif pour trois raisons. Tout d'abord la situation pandémique a empêché l'accès des chercheurs aux classes et aux pratiques réelles des enseignants. Nous avons dû changer notre méthodologie et travailler uniquement avec les enseignants. Ensuite, les enseignants participants étaient majoritairement des débutants en programmation. Nous avons donc analysé l'émergence de leurs pratiques déclarées et non pas des pratiques réelles établies. Dans ces circonstances, il n'était pas pertinent de dégager des critères d'efficacité. Enfin, le changement de méthodologie nous a amenés à créer des ressources avec les enseignants plutôt que d'analyser des ressources existantes.

Hypothèse de recherche

Notre hypothèse principale est que les ressources jouent un rôle primordial dans l'évolution des pratiques et l'introduction du numérique (Adler, 2000). Elles constituent l'ancrage privilégié des pratiques pédagogiques, aussi bien pour l'enseignement des mathématiques que pour la formation des enseignants. Dans la lignée d'Adler, nous adoptons une acceptation relativement large du concept de ressource, en y incluant tout ce qui permet de re-sourcer l'activité de l'enseignant, que ce soit du point de vue matériel, humain, culturel ou social. Cette acceptation nous paraît particulièrement bien adaptée au contexte de l'intégration de la programmation informatique. Nous partageons avec Adler (2000) l'idée que « la formation des enseignants doit porter plus

d'attention aux ressources, à ce qu'elles sont et à leur fonctionnement comme « extension » du professeur dans la pratique scolaire ». La conception et la mise en œuvre de ressources pédagogiques intégrant la programmation informatique constituent le moteur de notre recherche.

Partie B—Méthodologie

Revue de littérature (objectif 1)

Constitution du corpus de texte

Les textes constituant notre corpus ont été sélectionnés à l'aide des mots clés *programmation informatique, algorithmique, codage, robotique, enseignement des mathématiques*. La combinaison de ces différents mots clés nous a menés à une première sélection d'articles, rédigés en anglais, français ou italien, dans les bases de données bibliographiques : Google Scholar, ERIC, HAL et SOFIA. Parmi ces articles nous avons retenu uniquement ceux qui traitaient explicitement de l'utilisation de la programmation informatique dans l'enseignement des mathématiques, au primaire ou au secondaire. Nous avons ensuite utilisé les mots clés présents dans les résumés de ces articles pour compléter la base de données. 100 articles ont ainsi été retenus. Un travail de nettoyage du corpus a alors été mené. La lecture des titres et des résumés a permis de vérifier que les articles retenus concernaient bien l'enseignement des mathématiques, à la fin du primaire (K-11 et plus) et au secondaire. Les articles portant sur le lycée, en France, ou la High School ont également été conservés, à condition de porter sur des concepts mathématiques intéressants pour l'enseignement secondaire. Enfin, les articles proposant des expériences avec les enseignants ou les futurs enseignants de mathématiques ont été conservés également. Le corpus final comporte 69 textes (Annexe 1)

Analyse à l'aide du logiciel NVivo

Un codage a été réalisé à l'aide du logiciel N Vivo. Ce logiciel permet d'annoter un ensemble de documents à partir d'une liste de catégories, appelées nœuds, établie par les analystes. Un nœud est un ensemble de références relevant d'une même catégorie. Le codage se réalise en plusieurs étapes. Le corpus est parcouru une

première fois, et les extraits jugés pertinents sont associés à des nœuds prédéfinis. Un même extrait peut être associé à plusieurs nœuds. Lors de cette première étape, les nœuds sont libres, sans lien logique entre eux. Des nœuds peuvent être ajoutés en cours de codage pour nuancer une catégorie ou en créer une nouvelle. À l'issue de la phase de codage, les nœuds sont convertis en nœuds hiérarchiques et organisés sous forme d'arborescence, avec en tête des catégories générales/nœuds-parents, sous divisées en catégories particulières/nœuds-enfants (Annexe 2). Ce codage permet d'obtenir une vue d'ensemble du corpus, non plus structurée par les documents (les articles) mais par les nœuds/catégories (Annexe 3). Cette classification nous a permis de mettre au jour trois dimensions structurant l'ensemble des textes : la dimension informatique, la dimension mathématique et la dimension sociale.

Retour aux textes

Le logiciel Nvivo permet d'obtenir la liste des textes du corpus, classés selon le nombre d'extraits encodés dans chacune des catégories. Nous avons ainsi constitué des listes de lecture de 20 articles, rassemblant les textes les plus représentatifs de chaque dimension. Cette sélection a été enrichie avec des articles parus en 2022 et 2023, qui n'étaient pas dans la base de données initiale. Chacun des textes retenus a ensuite fait l'objet d'une lecture complète et d'une fiche de lecture, afin de brosser un portrait plus détaillé des problématiques de recherche en lien avec l'utilisation de de la programmation informatique dans l'enseignement des mathématiques.

Protocole de recherche (objectifs 2 et 3)

Nous avons mis en place une recherche qui s'apparente à une recherche-action. Nous avons opté pour une forme de recherche participative où les rapports entre praticiens et chercheurs forment le cœur de la démarche (Guillemette & Savoie-Zajc, 2012). Nous avons cherché, dans la lignée de (Guillemette, 2011), à développer un modèle d'accompagnement collectif soutenant l'ajustement des pratiques vers une intégration de la programmation informatique dans l'enseignement des mathématiques. La conjecture difficile dans le milieu éducatif a compliqué le processus de recrutement. Nous avons d'abord souffert des restrictions dues aux

mesures sanitaires qui nous ont empêchés d'aller dans les classes. La pénurie d'enseignants qui a suivi a eu pour conséquence de nous priver de participants, pour cause de surcharge de travail. Nous avons donc travaillé avec le même groupe de 12 volontaires, 10 femmes et deux hommes, tout au long du projet. Notre méthodologie a comporté deux phases bien distinctes.

1. Phase 1—2021/2022 : Le contexte postpandémique nous a conduits à adopter un fonctionnement classique alternant des journées de formation et de réflexion sur la programmation dans la plateforme Scratch et avec les robots EV3, et des périodes libres de conception et d'expérimentation de tâches pour la classe. La très grande hétérogénéité des niveaux en programmation de nos participants a rendu les moments de formation en grand groupe peu efficaces. Le besoin de partage exprimé par les participants nous a amenés à opter pour une démarche différente, inspirée des Lesson study, et basée sur le partage d'expérience.
2. Phase 2 – 2022/2024 : L'approche des Lesson study consiste à permettre à des chercheurs et des praticiens de travailler ensemble à la résolution de problèmes d'enseignement-apprentissage (Martin & Clerc-Georgy, 2017). Les conditions de mise en place du projet ne nous ont pas permis de mener une démarche complète de Lesson study mais avons repris l'idée de provoquer un changement de posture, aussi bien des chercheurs que des enseignants, à travers un partage des savoirs (Clerc-Georgy & Clivaz, 2016). Cela nous a amené vers une forme plus collaborative de transmissions des connaissances. Nous avons pour cela travaillé en deux temps : une analyse collective de tâches de programmation informatique proposées par les chercheurs puis la construction, en équipe de deux ou trois, d'une tâche pour la classe. Les tâches construites ont ensuite été expérimentées en classe par chacun des enseignants de l'équipe. Les équipes se sont rencontrées plusieurs fois pour préparer les tâches mais

aussi pour tirer le bilan des expérimentations. Certaines activités ont ainsi évolué au fil de leur mise en œuvre dans les différentes classes. Chaque équipe a ensuite présenté ses expériences au grand groupe. Une discussion collective a permis de tirer le bilan de l'ensemble des initiatives.

Données et analyses

La composition du groupe, très majoritairement féminin, a rendu caduque toute tentative d'analyse différenciée selon le sexe (ADS). N'ayant pas eu un accès direct aux élèves, nous avons dû renoncer aux questionnaires prévus dans le devis de recherche. Des questionnaires ont été passé auprès des enseignants à chacune des réunions bilans pour mesurer leur sentiment d'efficacité personnelle. Toutes les rencontres, aussi bien en grand groupe qu'en équipe, ont été enregistrées en audio. Les expérimentations en classe ont fait l'objet de captations vidéo. Nous avons recueilli les documents textuels et les programmes informatiques produits par les enseignants. Nous avons dégagé de ces données les gestes documentés observables (Gueudet & Trouche, 2008). Ces gestes sont des actions observables organisées par la structure des invariants opératoires de la pratique enseignante. Ils constituent la partie visible de cette structure qui se traduit sous forme de régularités observables dans l'activité du professeur. Les résultats présentés dans ce texte s'appuient sur les gestes documentés déclarés, c'est-à-dire ceux qui nous avons fait émerger d'une analyse du discours des enseignants lors des différentes rencontres. Des analyses complémentaires sont en cours pour dégager les gestes documentés observés en classe.

Partie C—Résultats

Revue de littérature

Dimension informatique

Cette dimension s'inscrit dans une perspective transversale et pluridisciplinaire. Elle explore les enjeux liés à l'introduction de la programmation informatique à l'école. Trois pôles principaux s'en dégagent : le rôle donné

à la programmation informatique à l'école, les différentes approches utilisées pour l'aborder et la place donnée à la résolution de problèmes. Le rôle donné à la programmation informatique est abordé selon différents angles: développement de la pensée (mathématique, algorithmique, informatique), outil pour l'enseignement de certains concepts (comme la valeur de position ou les fractales) ou pour réaliser des explorations mathématiques plus libres et enfin développement de la capacité à généraliser. La résolution de problèmes est abordée majoritairement dans une perspective mathématique, bien que 13% des références la mentionnent comme une compétence transversale que la programmation informatique peut permettre de développer. Elle est parfois mise en lien avec la pensée computationnelle. Ainsi 23 des 53 articles du corpus qui parlent de la résolution de problèmes aborde également le thème de la pensée computationnelle.

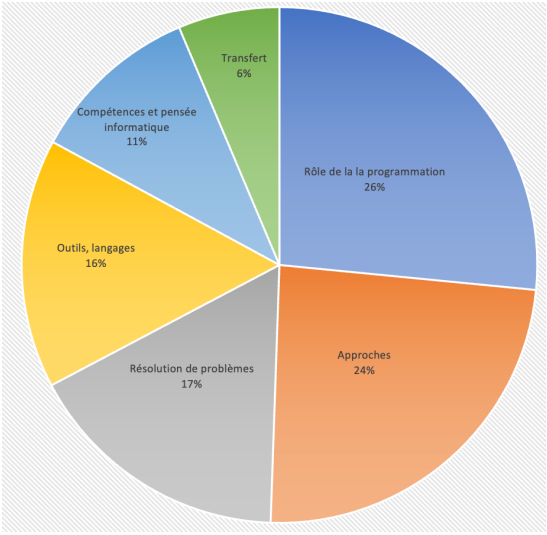


Figure 1. Répartition des références dans la dimension informatique.

Les auteurs décrivent également différentes approches intéressantes pour utiliser la programmation informatique en classe. Parmi les diverses approches deux dominent: le constructionnisme et l'approche algorithmique qui représentent respectivement 55% et 38% des références codées dans cette catégorie. L'approche constructionniste défend l'idée de permettre aux élèves d'explorer et de construire des idées par eux-mêmes. Certains auteurs donnent une place centrale à la mise en place d'activités qui permettent de s'assurer que les élèves rencontrent réellement les idées puissantes que l'environnement de programmation met à leur portée. D'autres insistent davantage sur la liberté d'exploration qui sous-tend toute approche

constructionniste. L'approche algorithmique consiste à donner une place centrale à la construction ou l'étude d'algorithmes, qui peut se faire en lien avec les enjeux de développement d'une pensée algorithmique, ou informatique. Les résultats de cette catégorie incitent à réfléchir aux liens entre la mise en œuvre d'activités de programmation informatique et le développement de la pensée critique, de la pensée logique, de la capacité à généraliser ainsi qu'à la place donnée à l'erreur.

Dimension mathématique

Cette dimension regroupe tous les nœuds qui concernent l'utilisation de la programmation informatique au service du développement des connaissances et des compétences mathématiques. Les éléments les plus saillants de cette catégorie sont les concepts mathématiques que l'on peut aborder par la programmation informatique qui représentent 20% des références codées (RC) dans cette catégorie. L'algèbre et la géométrie sont les domaines de prédilection de la mise en oeuvre de la programmation informatique au secondaire. Il est à noter que 13 articles du corpus considèrent l'algorithme comme un objet ou un concept mathématique, ce qui représente 28% des RC. Les résultats de cette catégorie incitent donc à réfléchir au lien entre pensée algorithmique et pensée mathématique, et plus particulièrement entre pensée algorithmique et pensée algébrique.

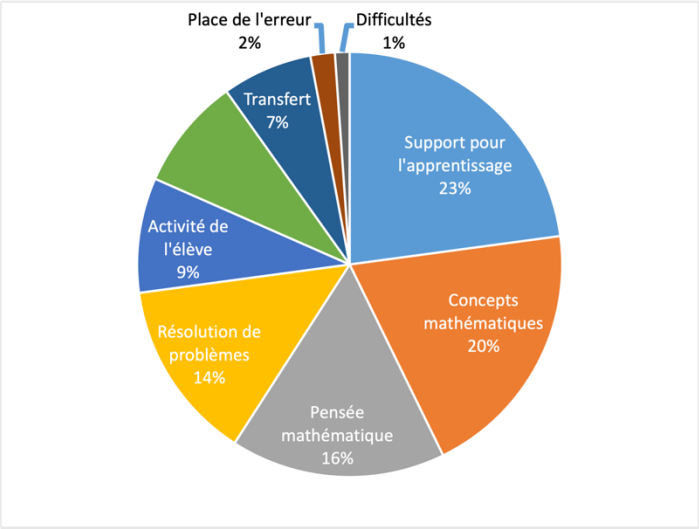


Figure 2. Répartition des références dans la dimension mathématique.

Dimension sociale.

Cette dimension rassemble tous les nœuds qui relèvent des enjeux sociaux et émotionnels en apprentissage que ce soit par la collaboration ou le partage. L'engagement des élèves est en effet une des grandes motivations pour introduire la programmation informatique dans l'enseignement et l'apprentissage des mathématiques. Ce nœud regroupe 42% des références encodées dans cette catégorie. Au sein de cette catégorie, divers aspects de l'engagement sont envisagés. 35% des références codées dans la catégorie engagements (RCE) s'intéressent à l'expérience de programmation des élèves sans élaborer sur la nature (mathématique, informatique ou transversale) de leurs activités. Dans le même ordre d'idée, 20% des RCE concernent la motivation au sens large, c'est-à-dire l'enthousiasme des élèves pour les tâches de programmation proposée. Le reste des RCE se partagent entre la mention d'un engagement dans une activité informatique (17%) et la mention d'une activité mathématique (22%). 6% des références concernent la fréquentation d'un concept mathématique précis par les élèves.

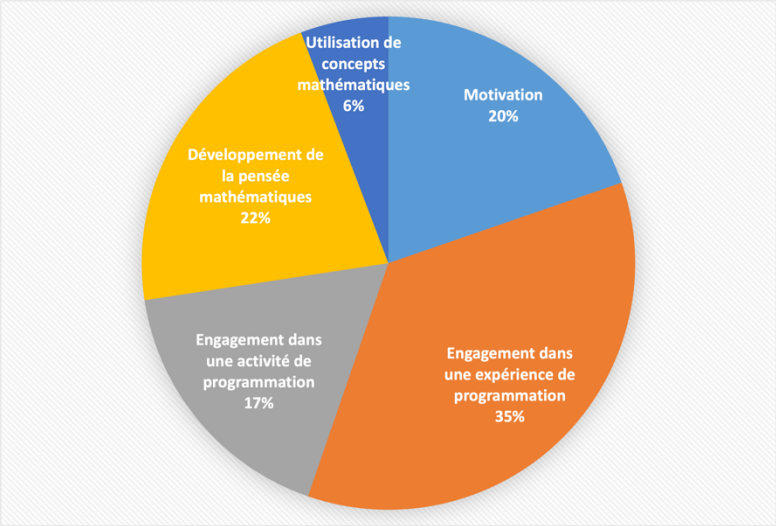


Figure 3. Répartition des références dans la sous-catégorie « Engagement et motivation » de la dimension mathématique.

La dimension sociale comprend aussi les préoccupations relatives au temps (17% des RC). La majorité de ces références relèvent le fait que les activités de programmation nécessitent du temps : du temps pour les élaborer, du temps pour les mettre en place mais aussi et surtout du temps à donner aux élèves pour qu'ils aient le temps de développer leur genèse instrumentale, d'explorer, d'établir les algorithmes nécessaires à la résolution de

problèmes. La dimension temporelle rejoint celle de la place donnée à l'erreur : les erreurs, et leurs corrections, sont partie intégrante du processus d'apprentissage par la programmation informatique. Il est donc fondamental de donner aux élèves le temps nécessaire à la détection et au traitement de ces erreurs. D'autres aspects positifs du recours à la programmation informatique sont abordés, comme le développement de la créativité ou le fait de favoriser la collaboration et le partage.

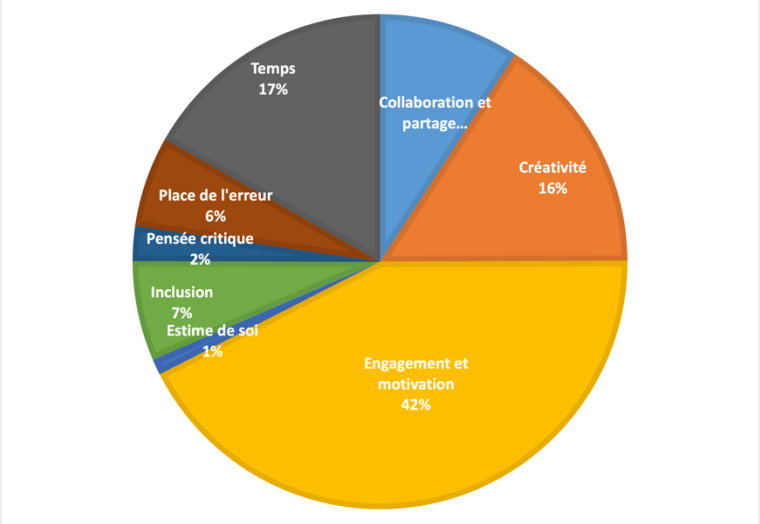


Figure 4. Répartition des codes dans la dimension sociale.

Principaux résultats de recherche

Nos résultats portent sur l'évolution des pratiques des enseignants relativement à l'intégration pédagogique de la programmation informatique. Nous les avons analysés selon les trois dimensions mises au jour dans notre revue de littérature. Les gestes documentés des enseignants et les intentions didactiques des enseignants sont catégorisés selon qu'ils concernent le développement de compétences purement informatique et/ou la construction d'un instrument d'apprentissage en mathématiques. La dimension sociale est également prise en compte.

Face à la très grande hétérogénéité des expertises en programmation des participants, nous avons rapidement renoncé aux formations magistrales. Nous avons alors adopté une démarche, inspirée des Lesson study, basée sur la construction de ressources pour la classe et le partage d'expérience. Cette démarche a porté ses fruits,

notamment en termes d'évolution des pratiques. Au démarrage du projet, tous les participants s'inscrivaient essentiellement dans les dimensions informatiques et sociales, considérant la programmation comme une activité importante socialement, utile pour la gestion de classe et favorisant l'engagement des élèves, mais à la marge de la planification annuelle des apprentissages. La programmation était majoritairement perçue comme une activité ludique comportant, intrinsèquement mais de façon diffuse, des liens avec les mathématiques et les sciences. Cette conception est conforme à celle prônée par les instances éducatives. Le guide de l'usage pédagogique de la programmation informatique publié par le ministère de l'Éducation considère, par exemple, la programmation informatique comme un contexte riche pour développer plusieurs dimensions de la compétence numérique, comme la production de contenu, la résolution de problèmes, la créativité et l'innovation. Elle est soutenue également par le discours marketing des développeurs de plateforme. Par exemple, la première phrase de présentation de la plateforme Scratch stipule que « Scratch est un langage de codage doté d'une interface visuelle simple qui permet aux jeunes de créer des histoires, des jeux et des animations numériques. »

Notre projet a amené les enseignants à se dégager peu à peu de l'injonction du ludique pour explorer les liens possibles avec les mathématiques, tout en maintenant l'engagement et la créativité à travers la création de sens. Les tâches proposées aux élèves ont été conçues de façon à inclure des apprentissages mathématiques prescrits par le PFEQ. La question de l'initiation des élèves à la programmation est cependant restée centrale. Elle a majoritairement été prise en charge en marge du travail mathématique. Les enseignants ont fait preuve de beaucoup de créativité dans leurs stratégies pour pallier leur manque d'expertise en programmation ainsi que celui de leurs élèves. Ainsi, malgré les lacunes en programmation, les activités ont été menées jusqu'au bout et la majorité des élèves y a pris plaisir. Les objectifs mathématiques n'ont cependant pas toujours été atteints. Souvent, le manque de recul algorithmique a empêché l'enseignant d'anticiper les enjeux mathématiques. Certains contenus mathématiques ont également été découverts en cours de route.

Le projet a fait émerger deux enjeux majeurs pour les enseignants : la possibilité de faire des liens explicites avec les apprentissages mathématiques prescrits par le programme et la quasi-impossibilité de reprendre une activité déjà existante. La confrontation à la résolution de problèmes algorithmique et informatique est incontournable, peu importe le niveau d'aisance en programmation. C'est tout le processus de planification qui doit donc être revisité à l'aune de cette nouvelle contrainte. Ce changement de posture s'ancre fortement dans le travail collaboratif proposé par le projet, mais aussi par les possibilités de soutien et d'accompagnement qu'il offre. Notre étude nous a permis de dégager quatre résultats principaux :

1. **Le potentiel pédagogique de la programmation informatique en classe de mathématiques**

   Les activités pédagogiques créées et expérimentées par les enseignants ont permis de mettre l'activité algorithmique au service des apprentissages mathématiques. Différents angles sont possibles : l'exploration de nouveaux processus (opérations arithmétiques, addition de fractions, constructions de figures…), l'exploration sous un nouveau jour d'un concept connu (nombre décimaux ou rationnels, figures géométriques, fonctions et fonctions réciproques…) ou encore le développement de la pensée algébrique (résolution d'équation, généralisation…). La programmation informatique permet d'aborder la résolution de problèmes mathématiques d'une façon alternative à la résolution papier-crayon. Elle offre de nouvelles façons de verbaliser et de partager les stratégies de résolution. Certains élèves connus pour leurs difficultés dans les résolutions traditionnelles se sont révélés être d'excellents raisonneurs dans un contexte de programmation. Cela ouvre également des pistes pour la différenciation des apprentissages.

2. **La nécessité d'un changement de posture avant, pendant et après la classe**

   Tous les participants au projet témoignent d'un changement de posture dans la gestion des interactions et de la dynamique de la classe. Cette posture les amène à assumer le fait que

leurs élèves apprennent plus vite qu'eux en programmation. Paradoxalement, ils expriment une plus grande confiance en eux car ils se sentent les garants des enjeux mathématiques. Il en résulte un réel plaisir à utiliser la programmation en classe, dans une reconnaissance du droit à l'erreur pour les élèves comme pour l'enseignant. Le groupe devient alors une ressource importante pour comprendre et corriger l'erreur.

3. La nécessité de coconstruire par l'expérience et le partage

Face aux difficultés et au manque d'expertise, la communauté constitue une ressource incontournable. Le travail collaboratif est à favoriser à tous les niveaux : pour les élèves, en groupe classe, pour les enseignants au sein de communautés pédagogiques et professionnelles locales et élargies. Ce travail collaboratif doit comporter une part de mise en action et des retours d'expérience. Il encourage et soutient la diffusion et surtout la reconnaissance, financière et intellectuelle, des initiatives portées par tous les intervenants.

4. La nécessité de baliser clairement les compétences et les habiletés en programmation informatique

Ni les enseignants, ni les élèves ne sont formés en programmation informatique, il est donc difficile pour les praticiens de penser l'articulation entre travail mathématique, travail informatique et travail algorithmique (Briant & Bronner, 2015; Venant et al., 2022). Les enseignants doivent assurer l'initiation informatique, la leur et celles de leurs élèves en même temps que sa mise en œuvre mathématique. C'est à la fois épuisant et contre- productif. Ils doivent pouvoir s'appuyer sur des connaissances établies chez les élèves. Il faudrait pour cela que la programmation informatique ne soit plus une activité laissée à la libre initiative de l'enseignant, mais une activité prescrite par le curriculum.

Il en ressort que la programmation informatique n'est pas un simple outil pédagogique. C'est une activité dont les aspects algorithmiques et techniques doivent être documentés et prescrits pour pouvoir être articulés de façon pertinente avec les apprentissages disciplinaires.

Retombées

Le projet a permis de confirmer le potentiel de l'articulation des pensées informatique et mathématique. Il a montré également que cette articulation demande des conditions favorables précises pour sa mise en œuvre effective. Les enjeux techniques (connaître les différents blocs/instructions, s'initier par le défi et le jeu) et ludiques sont actuellement prioritaires. Le projet montre la nécessité de déplacer la priorité vers des enjeux plus didactiques, centrés sur le développement de compétences. Il devrait en résulter une emphase moins grande sur les outils d'initiation à la programmation au profit de questionnements sur la posture à adopter pour guider les élèves dans un travail simultanément informatique et mathématique. L'articulation entre ces deux processus et la possibilité d'enrichissement mutuel reste encore très diffuse dans les politiques et documents ministériels. L'obstacle majeur vers cette évolution est propre au contexte québécois. Les enseignants ne sont pas formés en programmation informatique, il leur est donc difficile de penser l'articulation entre travail mathématique, travail informatique et travail mathématique. Or c'est de cette articulation que dépend l'analyse didactique des instruments d'apprentissage formés…. Par ailleurs, cette genèse instrumentale didactique (Venant, 2023) est aussi entravée par le manque de formation des élèves. On pense souvent que le facteur limitant dans cette problématique est le temps. Sans sous-estimer le côté chronophage de l'apprentissage de la programmation, nous pouvons dire que notre projet montre que le partage et le soutien de la communauté constituent des voies très prometteuses pour contourner cet obstacle. Nos résultats montrent que le groupe constitue une ressource précieuse, à condition d'y permettre le partage d'expériences significatives et abouties (Dewey, 1938). Le travail collaboratif est à favoriser à tous les niveaux :

- Groupe classe, travail en dyade, appui sur des élèves ressources

- Communauté pédagogique de l'école : co-enseignement, travail en équipe, soutien pédagogique en classe
- Communauté de pratique élargie : avec des enseignants d'autres écoles, d'autres niveaux mais aussi des personnes-ressources expertes.
- Communauté professionnelle institutionnelle : importance de la valorisation, de la diffusion, de la reconnaissance.

Contributions en termes d'avance des connaissances

D'un point de vue pédagogique, le projet a permis la rédaction d'un guide d'initiation à la programmation informatique intitulé « Apprendre à programmer avec les mathématiques. » qui sera publié par les éditions JFD dans le courant de l'année 2025.

D'un point de vue théorique, le projet a permis de mieux comprendre les interactions entre le travail algorithmique, sous-jacent à l'activité de programmation, et le travail mathématique. La revue de littérature que nous avons menées en début de projet a confirmé que même si ces interactions sont connues depuis les prémisses de la programmation informatique, elles ont été peu explorées. Il ressort de notre étude que la programmation informatique constitue un outil indéniable pour l'enseignement des mathématiques. Cependant, beaucoup de recherches restent à faire à propos de sa mise en œuvre au service du développement des connaissances scientifiques, ainsi que sur les transferts possibles des compétences informatiques vers les compétences mathématiques. Deux enjeux principaux se dégagent cependant : celui de la place de l'algorithme, entre mathématique et informatique, et celui de la formation des enseignants. Cette formation doit prendre en charge l'apprentissage de la programmation mais aussi les modalités d'intervention en classe ainsi que la conception d'activités qui soutiennent le développement des perspectives et pratiques de programmation.

D'un point de vue à la fois méthodologique et conceptuel, notre projet met en lumière la force des approches expérientielles au sein de communautés de pratique. Nos choix méthodologiques ont permis à tous les acteurs de notre projet, élèves, enseignants, conseillers pédagogiques, étudiants et chercheurs, de vivre des

expériences informatiques et humaines, au cours desquelles ils se sont développés personnellement et professionnellement (le cas échéant). Nous rejoignons ainsi les travaux menés dans le courant de l'approche expérientielle (Dewey, 1938; Kolb, 2014; Lewin, 1951). L'apprentissage y est défini comme un processus visant l'intégration des dimensions cognitives, socioaffectives et comportementales dans un processus d'éducation et de formation de la personne. L'apprenant acquiert de nouvelles connaissances en se mettant en action et en s'engageant aussi bien cognitivement qu'affectivement à la lumière de ses expériences passées (Van Nieuwenhoven & Cividini, 2016). Notre projet a mis en évidence la dimension expérientielle de la programmation informatique. Nous avons pu montrer qu'elle permet de mettre en œuvre les 5 étapes du modèle d'apprentissage expérientiel décrites par Van Nieuwenhoven and Cividini (2016) :

1. Elle permet l'ouverture à l'apprentissage et invite l'apprenant à prendre contact avec l'objet d'apprentissage.
2. Elle permet un engagement actif dans la démarche d'apprentissage, en rendant tangible des idées abstraites, comme la généralisation d'un motif, ou la visualisation de forme géométrique.
3. Elle favorise l'identification, c'est-à-dire le moment de s'approprier l'apprentissage d'un point de vue intellectuel et émotionnel. À travers ses interactions avec la plateforme de programmation, l'apprenant porte également un regard critique sur son action et celles de ses pairs.
4. L'étape qui reste à construire dans le contexte scolaire est celle de l'intériorisation. Elle se manifeste à travers le transfert des compétences acquises en contexte de programmation vers les autres contextes ou disciplines.
5. La cinquième étape, celle de la dissémination a été atteinte par tous les participants au projet. Tous ont entrepris de témoigner de leurs apprentissages et des changements qui se sont opérés dans leurs pratiques professionnelles. Ces témoignages prennent différentes

formes depuis la création d'un site web de partage de ressources et d'expériences (Annexe 4), jusqu'à l'animation d'ateliers dans des colloques professionnels.

Partie D – Pistes de solution ou d'action

Nos résultats montrent que l'expérience, le partage et la reconnaissance (financière mais aussi institutionnelle) sont les trois piliers sur lesquels peut reposer une intégration effective de la programmation informatique à l'école. Les pistes de solutions que nous proposons sont donc les suivantes :

S'appuyer sur le curriculum

La programmation informatique ne trouvera pas sa place en tant que pratique pédagogique tant qu'on n'aura pas défini sa place en tant que contenu à enseigner. Il ne s'agit pas, en effet, d'un outil qu'on peut choisir d'utiliser de façon ponctuelle. La pratique de la programmation informatique repose sur le développement d'une pensée algorithmique et informatique qui demande l'acquisition de concepts clés. Tant que des savoirs de base comme *entrées et sorties*, *séquences*, *instructions*, *variables*, *structures de contrôle* et *instructions conditionnelles* restent tacites et partiellement acquis, il est irréaliste de penser un transfert des apprentissages informatiques vers les disciplines scolaires. L'enseignant qui veut construire un instrument pédagogique sous forme de programme informatique doit pouvoir savoir quelles sont les connaissances de ses élèves sur lesquelles ils peuvent s'appuyer. Tous ces concepts, et les processus qu'ils mettent en jeu (affectation, raisonnement itératif, généralisation…) doivent être clairement décrits et situés dans une progression des apprentissages.

Puiser dans la force du collectif.

La mise place des communautés de pratique, ou des groupes d'échange d'expérience, rassemblant des enseignants de différentes écoles au sein d'un même centre de services est un mode d'action à privilégier pour

le développement de la pensée informatique et algorithmique ainsi que pour rendre possible un recul didactique sur ces pensées. En rassemblant des enseignants de différents niveaux d'enseignement, on favorise du même coup les transitions inter ordres ou inter cycles. Les pratiques doivent comporter une large part expérientielle autour de la résolution collaborative de problèmes de programmation informatique et de la co-construction d'activités pédagogiques, éventuellement en équipe. Les partages doivent porter sur toute la temporalité de la pratique enseignante : avant, pendant et après la classe.

Placer l'expertise au cœur du processus.

Les experts doivent être des membres à part entière de la communauté de pratique, au même titre que les autres participants. Ils doivent partager leurs expériences, et ne pas rester en retrait même s'ils sont les garants de la co-émergence des savoirs. Ils sont partie prenante des discussions et doivent accepter de voir leurs propres conceptions évoluer. Leur place n'est pas nécessairement dans la classe. Cependant, un accompagnement ponctuel, sous forme de co-enseignement, n'est pas exclu et peut rassurer certains participants hésitants à se lancer.

Encourager la formation continue par la dissémination.

L'engagement dans une communauté de pratique de type expérientiel aboutit naturellement à des initiatives de dissémination. Elles prennent différentes formes et s'adressent à différents publics. Elles peuvent être locales, au sein d'une école, quand un participant accueille dans sa classe un collègue curieux d'en savoir plus, ou quand un co-enseignement se met en place avec d'autres collègues. Elles peuvent se situer au niveau du centre de services scolaire, par la diffusion des ressources, des partages d'expérience, ou l'animation de formations continues internes. Enfin, elles peuvent s'ouvrir à la communauté professionnelle entière, par l'animation d'ateliers dans des colloques

professionnels ou la collaboration avec des partenaires extérieurs. Toutes ces initiatives devraient être encouragées et soutenues officiellement par l'institution.

Ces propositions s'appuient sur les résultats de notre projet de recherche. Il convient cependant de souligner que ces résultats portent sur un petit échantillon d'enseignants de la fin du primaire et du secondaire, tous volontaires pour intégrer la programmation informatique. Nos expériences antérieures nous donnent à penser que cet échantillon est très représentatif de l'ensemble des enseignants de mathématiques québécois qui manifestent un intérêt pour intégrer la programmation informatique dans leur pratique, même s'ils débutent dans ce domaine. Notre méthodologie peut, sous certaines conditions portant sur la composition de la communauté de pratique, être généralisée à des enseignants n'ayant a priori aucune affinité avec cette approche.

Partie E – Nouvelles pistes ou question de recherche

Beaucoup de recherches restent à faire à propos de la mise en œuvre de la programmation informatique au service du développement des connaissances mathématiques, ainsi que sur les transferts possibles des compétences informatiques vers les compétences mathématiques. Les mathématiques de fin de secondaire sont particulièrement peu explorées. Deux enjeux principaux se dégagent: celui de la place de l'algorithme, entre mathématique et informatique, et celui de la formation des enseignants. La méthodologie que nous avons mise en place constitue une piste de réponse que nous aimerions explorer davantage, à plus grande échelle, en intégrant des enseignants a priori peu disposés à utiliser la programmation informatique en classe. D'autres modalités sont à penser aussi bien en formation continue qu'initiale. Il faut continuer d'étudier le rôle que peut jouer la conception d'activités pédagogiques comme médiateur aussi bien de l'apprentissage de la programmation que des postures d'intervention en classe. Cela passe par un renforcement du dialogue entre la didactique des mathématiques et l'algorithmique. Il est important, en particulier, d'expliciter les processus algorithmiques sous-jacents et les différents types de pensées que les activités de programmation permettent

de développer (Venant, 2022). Dans la lignée de Bråting & Kilhamn (2021) nous appelons à davantage de recherches sur les différences entre les pensées mathématiques, algorithmique et informatique. Ces trois domaines sont essentiels lorsque l'on travaille avec les mathématiques, mais pour les enseigner, nous devons déterminer l'utilité spécifique de chacun d'entre eux (TEMA et al., 2023)

Partie F-- Bibliographie

Adler, J. (2000). Conceptualizing resources as a theme for teacher education. Journal of Mathematics Teacher Education, 3(3), 205-224.

Barma, S. (2018). Réaliser une étude de cas multiple qui vise à affiner les connaissances sur l'usage pédagogique ou didactique de la programmation dans les écoles du Québec - Rapport final. https://numerique.banq.qc.ca/patrimoine/details/52327/4075493?docpos=2

Bråting, K., & Kilhamn, C. (2021). Exploring the intersection of algebraic and computational thinking. Mathematical Thinking and Learning, 23(2), 170-185.

Briant, N., & Bronner, A. (2015). Étude d'une transposition didactique de l'algorithmique au lycée: une pensée algorithmique comme un versant de la pensée mathématique. Actes du Colloque EMF2015–GT3,

Clerc-Georgy, A., & Clivaz, S. (2016). Évolution des rôles entre chercheurs et enseignants dans un processus Lesson study: quel partage des savoirs. Le partage des savoirs dans les processus de recherche en éducation, 189.

Dewey, J. (1938). Experiential learning. New Jersey: Pentice Hall.

Gueudet, G., & Trouche, L. (2008). Du travail documentaire des enseignants: genèses, collectifs, communautés: Le cas des mathématiques. Éducation et didactique, 3, 7-33.

Guillemette, S. (2011). Étude de l'ajustement de pratiques vers une gestion différenciée de l'activité éducative chez les directions d'établissement d'enseignement: expérimentation d'un modèle d'accompagnement collectif (thèse de doctorat inédite). Université de Sherbrooke.

Guillemette, S., & Savoie-Zajc, L. (2012). La recherche-action et ses rapports de co-construction de savoirs et de formation dans une perspective de professionnalisation entre acteurs praticiens et chercheurs. Formation et profession, 20(3), 41-52.

Henda, M. B. (2017). L'enseignement du code informatique à l'école. Actes de la 5e rencontre annuelle d'ORBICOM.

Karsenti, T., & Bugmann, J. (2017). Enseigner et apprendre avec le numérique. Les Presses de l'Université de Montréal.

Kolb, D. A. (2014). Experiential learning: Experience as the source of learning and development. FT press.

Lewin, K. (1951). Intention will and need.

Romero, M., Lille, B., & Patiño, A. (2017). Usages créatifs du numérique pour l'apprentissage au XXIe siècle. PUQ.

TEMA., Guillemette, D., Jeannotte, D., Knoll, E., Passaro, V., Saboya, M., & Venant, F. (2023). Pensée algébrique et pensée algorithmique: réflexion autour d'une tâche de généralisation. Espace Mathématique Francophone, 2022, Cotonou, Bénin.

Van Nieuwenhoven, C., & Cividini, M. (2016). Quand l'étudiant devient enseignant: préparer et soutenir l'insertion professionnelle. Presses universitaires de Louvain.

Venant, F. (2022). Conception de tâches de programmation informatique pour l'enseignement des mathématiques : prendre en compte les enjeux algorithmiques. Rendez-vous didactiques, Paris.

Venant, F. (2023). Enjeux didactiques dans les genèses instrumentales professionnelles des futurs enseignants de mathématiques au Québec. Recherches en didactique des mathématiques, 43(1), 15-46.

Venant, F., Guillemette, D., Jeannotte, D., Knoll, E., Passaro, V., & Saboya, M. (2022). Articuler travail algorithmique et travail mathématique. ETM7: 7éme symposium sur les espaces mathématiques., Strasbourg.

Wing, J. (2011). Research notebook: Computational thinking—What and why. The link magazine, 6, 20-23.

Wing, J. M. (2008). Computational thinking and thinking about computing. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366(1881), 3717-372

Annexe 1 – Corpus pour la revue de littérature

Au, W. K., & Leung, J. P. (1991). Problem solving, instructional methods and Logo programming. *Journal of Educational Computing Research*, *7*(4), 455-467.

Balacheff, N. (2017). Seymour Papert (1928-2016) Aux sources d'une pensée innovante et engagée. *Recherches en didactique des mathématiques*, *37*(2/3), 383-396.

Bar-On, E., & Or-Bach, R. (1988). Programming mathematics: a new approach in introducing probability to less able pupils. *International Journal of Mathematical Education in Science and Technology*, *19*(2), 281-297.

Barroso, R., Castro, A., Rocha, A., th Iberian Conference on Information, S., & Technologies, C. (2018). Computer programming as a tool to improve mathematic skills in basic education. *Iberian Conference on Information Systems and Technologies, CISTI*, *2018-June*, 1-3. https://doi.org/10.23919/CISTI.2018.8399192

Batista, S. C. F., & Baptista, C. B. F. (2014). Learning object for linear systems: Scratch in mathematics. *International Journal on New Trends in Education and Their Implications*, *5*(1), 71-81.

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2016). Building mathematical knowledge with programming: insights from the ScratchMaths project.

Benton, L., Hoyles, C., Kalas, I., & Noss, R. (2017). Bridging primary programming and mathematics: Some findings of design research in England. *Digital Experiences in Mathematics Education*, *3*(2), 115-138.

Benton, L., Saunders, P., Kalas, I., Hoyles, C., & Noss, R. (2018). Designing for learning mathematics through programming: A case study of pupils engaging with place value. *International Journal of child-computer interaction*, *16*, 68-76.

Birebent, A., & Chi, T. N. (2005). Conception d'une ingénierie didactique pour l'introduction d'éléments d'algorithmique et de programmation dans l'enseignement secondaire des mathématiques.

Bizzarri, G., Forlizzi, L., & Proietti, G. (2013). Potpourri di tecnologie didattiche. *Proceedings of Didamatica*.

Bråting, K., & Kilhamn, C. (2021). Exploring the intersection of algebraic and computational thinking. *Mathematical Thinking and Learning*, *23*(2), 170-185.

Briant, N., & Bronner, A. (2015). Étude d'une transposition didactique de l'algorithmique au lycée: une pensée algorithmique comme un versant de la pensée mathématique. Actes du Colloque EMF2015–GT3,

Button, T. (2014). Programming in Mathematics A level. *Mathematics in School*, *43*(4), 4-5. http://www.jstor.org/stable/24767701

Calder, N. (2010). Using scratch: an integrated problem-solving approach to mathematical thinking. *Australian Primary Mathematics Classroom*, *15*(4), 9-14.

Capone, R., DEL SORBO, M. R., Musmarra, P., Veronesi, I., & Esposito, A. (2018). Coding e Pensiero Computazionale per il potenziamento delle competenze logiche e matematiche. *Nuovi metodi e saperi per formare all'innovazione AICA*, 155-159.

Clements, D. H., & Meredith, J. S. (1993). Research on Logo: Effects and efficacy. *Journal of Computing in Childhood Education*, *4*(4), 263-290.

De Corte, E. (1992). On the Learning and Teaching of Problem-solving Skills in Mathematics and LOGO Programming. *Applied Psychology: an international review*, *41*(4), 317-331.

Du Boulay, J. (1980). Teaching teachers mathematics through programming. *International Journal of Mathematical Educational In Science and Technology*, *11*(3), 347-360.

Felicia, A., & Sharif, S. (2014). A review on educational robotics as assistive tools for learning mathematics and science. *Int. J. Comput. Sci. Trends Technol*, *2*(2), 62-84.

Flores, A. (1985). *Effect of computer programming on the learning of calculus concepts* The Ohio State University].

Foerster, K.-T. (2016). Integrating programming into the mathematics curriculum: Combining scratch and geometry in grades 6 and 7. Proceedings of the 17th annual conference on information technology education,

Frassia, M. G. (2014). Esperienze di programmazione al computer:'punti di vista'sul calcolo della probabilità. *Mondo Digitale*, *13*(51), 726-735.

Freudenthal, E. A., Lim, K. H., Kranz, S., Tabor, C., & Ramirez, J. L. (2013). Using programming to strengthen mathematics learning in 9th grade algebra classes. 2013 ASEE Annual Conference & Exposition,

Gadanidis, G. (2017). Artificial intelligence, computational thinking, and mathematics education. *The International Journal of Information and Learning Technology*, *34*(2), 133-139.

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, *42*(1), 38-43.

Grover, S., Pea, R., & Cooper, S. (2015). Designing for deeper learning in a blended computer science course for middle school students. *Computer science education*, *25*(2), 199-237.

Haspekian, M., & Nijimbéré, C. (2016). Favoriser l'enseignement de l'algorithmique en mathématiques: une question de distance aux mathématiques? *Éducation et didactique*, *10*(10-3), 121-135.

Henderson, P. B. (1990). Discrete mathematics as a precursor to programming. Proceedings of the twenty-first SIGCSE technical symposium on Computer science education,

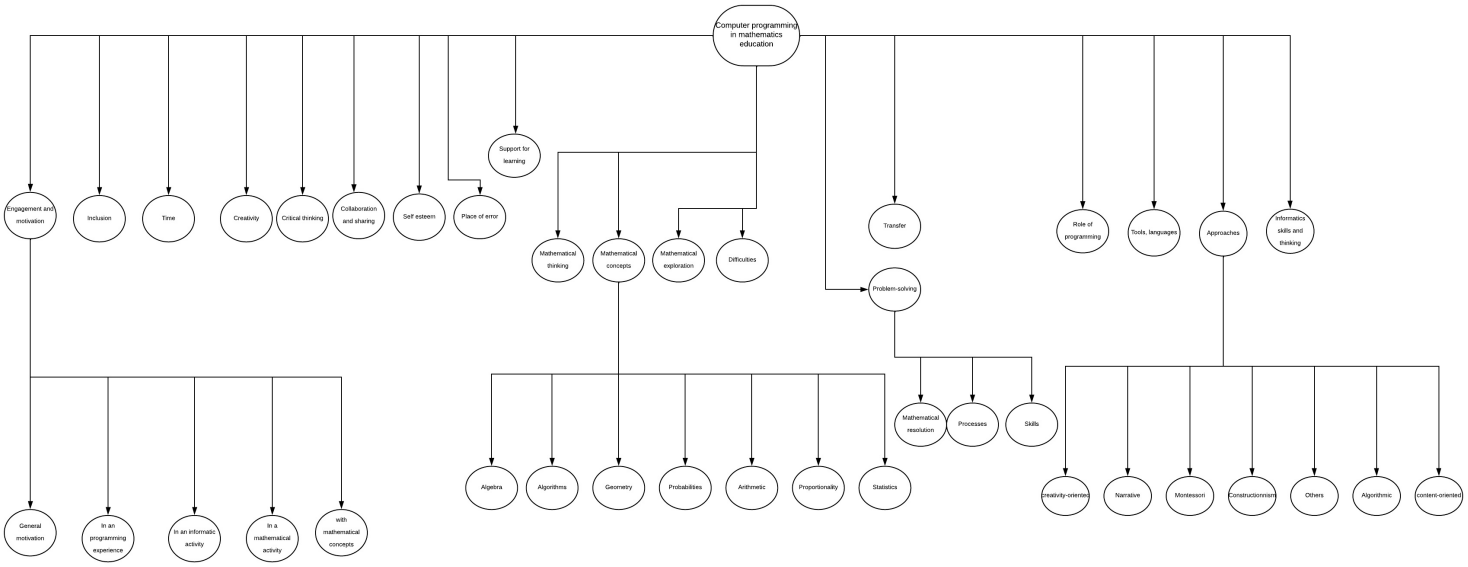
How, M.-L., & Looi, C.-K. (2018). Using Grey-Based Mathematical Equations of Decision-Making as Teaching Scaffolds: From an Unplugged Computational Thinking Activity to Computer Programming. *International Journal of Computer Science Education in Schools*, *2*(2), n2.

Hoyles, C. (1987). Tools for learning: Insights for the mathematics educator from a Logo programming environment. *For the Learning of Mathematics*, *7*(2), 32-37.

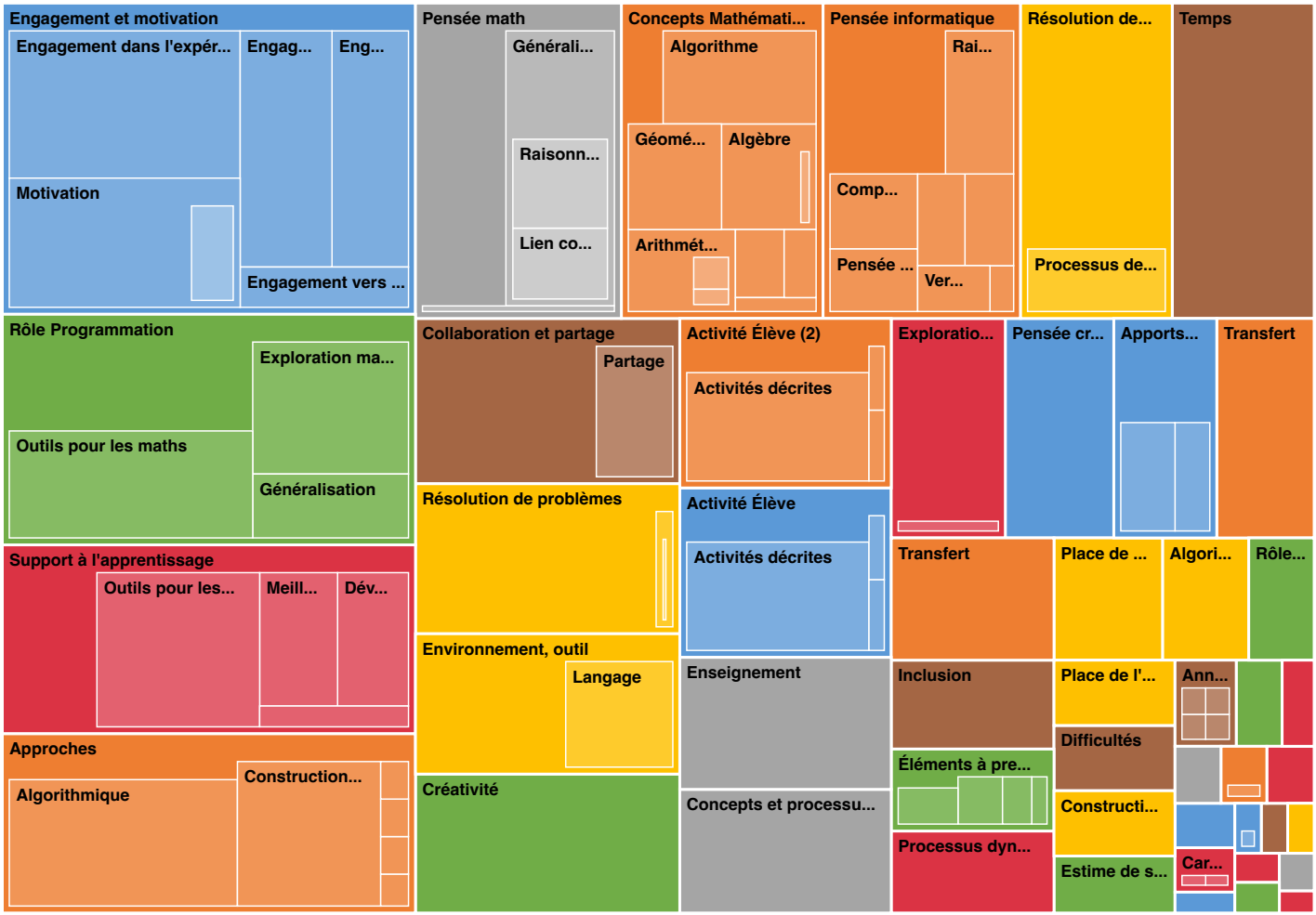
Hoyles, C., & Noss, R. (1987). Synthesizing mathematical conceptions and their formalization through the construction of a Logo-based school mathematics curriculum. *International Journal of Mathematical Education in Science and Technology*, *18*(4), 581-595.
Hsu, T.-C., Chang, S.-C., & Hung, Y.-T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education*, *126*, 296-310.
Iskrenovic-Momcilovic, O. (2020). Improving Geometry Teaching with Scratch. *International Electronic Journal of Mathematics Education*, *15*(2), em0582.
Järvinen, E.-M. (1998). The Lego/Logo Learning Environment inTechnology Education: An Experiment in a Finnish Context. *Journal of Technology Education*, *9*(2).
Johnson, D. C. (2000). Algorithmics and programming in the school mathematics curriculum: support is waning-is there still a case to be made? *Education and Information Technologies*, *5*, 201-214.
Kallia, M., van Borkulo, S. P., Drijvers, P., Barendsen, E., & Tolboom, J. (2021). Characterising computational thinking in mathematics education: a literature-informed Delphi study. *Research in Mathematics Education*, *23*(2), 159-187.
Kilhamn, C., Bråting, K., Helenius, O., & Mason, J. (2022). Variables in early algebra: exploring didactic potentials in programming activities. *ZDM–Mathematics Education*, *54*(6), 1273-1288.
Kynigos, C. (1993). Children's inductive thinking during intrinsic and Euclidean geometrical activities in a computer programming environment. *Educational Studies in Mathematics*, *24*(2), 177-197.
Kynigos, C., & Grizioti, M. (2018). Programming approaches to computational thinking: Integrating Turtle geometry, dynamic manipulation and 3D Space. *Informatics in Education*, *17*(2), 321-340.
Lagrange, J.-B. (2020). Algorithmics. *Encyclopedia of mathematics education*, 44-47.
Lagrange, J.-B., & Rogalski, J. (2017). Savoirs, concepts et situations dans les premiers apprentissages en programmation et en algorithmique. *Annales de Didactiques et de Sciences Cognitives.*
Laval, D. Une ingénierie didactique autour de "l'algorithme de dichotomie" illustrant un enseignement de l'algorithme en première scientifique comme objet d'apprentissage dans un champ des mathématiques: l'analyse. *APMEP.*
Lewis, C. M., & Shah, N. (2012). Building upon and enriching grade four mathematics standards with programming curriculum. *Proceedings of the 43rd ACM technical symposium on Computer Science Education*.
McCoy, L. P., & Burton, J. K. (1988). The relationship of computer programming and mathematics in secondary students. *Computers in the Schools*, *4*(3-4), 159-166.
Misfeldt, M., & Ejsing-Duun, S. (2015). Learning mathematics through programming: An instrumental approach to potentials and pitfalls.
Mor, Y., & Noss, R. (2008). Programming as mathematical narrative. *International Journal of Continuing Engineering Education and Life Long Learning*, *18*(2), 214-233.
Nguyen C. T, & Bessot, A. (2003). La prise en compte des notions de boucle et de variable informatique dans l'enseignement des mathématiques au lycée. *Petit x*, *62*.
Nordby, S. K., Bjerke, A. H., & Mifsud, L. (2022). Computational thinking in the primary mathematics classroom: A systematic review. *Digital Experiences in Mathematics Education*, *8*(1), 27-49.
Noss, R. (1986). Constructing a conceptual framework for elementary algebra through Logo programming. *Educational Studies in Mathematics*, *17*(4), 335-357.
Olive, J. (1991). Logo programming and geometric understanding: An in-depth study. *Journal for Research in Mathematics Education*, 90-111.
Ortiz, E., & MacGregor, S. K. (1991). Effects of logo programming on understanding variables. *Journal of Educational Computing Research*, *7*(1), 37-50.
Papert, S. (1996). An exploration in the space of mathematics educations. *Int. J. Comput. Math. Learn.*, *1*(1), 95-123.
Peelle, H. (1979). Teaching Mathematics via APL (A Programming Language). *The Mathematics Teacher*, *72*(2), 97-116.
Pérez, A. (2018). A framework for computational thinking dispositions in mathematics education. *Journal for Research in Mathematics Education*, *49*(4), 424-461.
Psycharis, S., & Kallia, M. (2017). The effects of computer programming on high school students' reasoning skills and mathematical self-efficacy and problem solving. *Instructional Science*, *45*(5), 583-602. https://doi.org/10.1007/s11251-017-9421-5
Rodríguez-Martínez, J. A., González-Calero, J. A., & Sáez-López, J. M. (2020). Computational thinking and mathematics using Scratch: an experiment with sixth-grade students. *Interactive Learning Environments*, *28*(3), 316-327.
Shafto, S. A. (1986). Programming for learning in mathematics and science. *ACM SIGCSE Bulletin*, *18*(1), 296-302.
Suters, L., & Suters, H. (2020). Coding for the core: Computational thinking and middle grades mathematics. *Contemporary Issues in Technology and Teacher Education*, *20*(3), 435-471.
Tchounikine, P. (2017). Initier les élèves à la pensée informatique et à la programmation avec Scratch. *Consulté à l'adresse http://lig-membres. imag. fr/tchounikine/PenseeInformatiqueEcole. html*, 1-36.
Thành, N. C., & Bessot, A. (2010). Introduire des éléments d'algorithmique et de programmation dans l'enseignment secondaire? Une ingénierie didactique. *Petit x*, *83*.
Venant, F. and Thibault, M. (2019). Programmer pour apprendre en probabilités. In V. Martin, Thibault, M., & Theis, L (Ed.), *Enseigner les premiers concepts de probabilités : un monde de possibilités !* Presses de l'Université du Québec.
Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, *20*, 715-728.
Wain, G., & Flower, S. (1992). Mathematics homework on a micro. *Mathematics in School*, *21*(3), 8-11.

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127-147.
Wilensky, U. (1995). Paradox, programming, and learning probability: A case study in a connected mathematics framework. *The Journal of Mathematical Behavior*, *14*(2), 253-280.
Wilkerson-Jerde, M. H. (2014). Construction, categorization, and consensus: Student generated computational artifacts as a context for disciplinary reflection. *Educational Technology Research and Development*, *62*, 99-121.
Winter, V., Love, B., & Corritore, C. (2018). The art of the Wunderlich cube and the development of spatial abilities. *International journal of child-computer interaction*, *18*, 1-7.
Yuana, R. A., Faisal, M., Pangestu, D., & Putri, Y. R. L. (2015). Math thematic learning through the introduction of basic science-based programming games virtual robot for high school students. *Advanced Science Letters*, *21*(7), 2235-2238.

Annexe 2—Hiérarchie des nœuds pour le codage dans Nvivo



Annexe 3- Vue du corpus structuré par les catégories/nœuds



Annexe 4-Site web réalisé par une participante au projet

Adresse : https://sites.google.com/cscapitale.qc.ca/programmation/accueil

Aperçu :