

# Cas d'utilisations de LLM par un enseignant-chercheur

Nicolas Delestre

# Activités d'un Enseignant Chercheur pouvant être aidées par un LLM

## Enseignement

- Produire des ressources pédagogiques
- Produire des ressources d'évaluation
- Répondre aux questions des étudiants

## Chercheur

- Étudier des articles
- Faire de la recherche : hypothèse, expérimentation, validation
- Publier

## Administratif

- Organiser : emploi du temps, stages étudiants, relations internationales, etc.
- Rédiger : appels à projet, enquêtes, rapports
- Représenter : conseils, comités, commissions, etc.

# Activités d'un Enseignant Chercheur pouvant être aidées par un LLM

## Enseignement

- **Produire des ressources pédagogiques**
- **Produire des ressources d'évaluation (x2)**
- Répondre aux questions des étudiants

## Chercheur

- **Étudier des articles**
- Faire de la recherche : hypothèse, expérimentation, validation
- Publier

## Administratif

- Organiser : emploi du temps, stages étudiants, relations internationales, etc.
- **Rédiger : appels à projet, enquêtes, rapports**
- Représenter : conseils, comités, commissions, etc.

# Activités d'un Enseignant Chercheur pouvant être aidées par un LLM

## Enseignement

- **Produire des ressources pédagogiques**
- **Produire des ressources d'évaluation (x2)**
- Répondre aux questions des étudiants

### Attention !

- Les copies d'écran proposées sont réelles avec donc des fautes de frappes, d'orthographe, etc.
- Je vais faire de l'anthropomorphisme. . .

## Chercheur

- **Étudier des articles**
- Faire de la recherche
- Publier

## Administratif

- Organiser : emploi du temps, stages étudiants, relations internationales, etc.
- **Rédiger : appels à projet, enquêtes, rapports**
- Représenter : conseils, comités, commissions, etc.

# Quels LLM j'utilise ?

## ChatBot

- ChatGPT, Gemini, Le chat
- Pour discuter, obtenir des réponses, obtenir des exemples de textes ou de codes, me corriger un texte, me proposer un plan

## Copilot

- Visual Code, Emacs
- Pour améliorer la productivité (code, texte, traduction, ...)

## NotebookLM

- Pour obtenir des réponses référencées à des questions sur un corpus

# Contexte : remédiation de cours pour la pédagogie inversée

Entrée et sorties standards

Trois flux standards sous UNIX orientés caractère

- 1 Entrée standard par défaut le clavier
- 2 Sortie standard par défaut l'écran, le terminal
- 3 Sortie d'erreur standard par défaut l'écran, le terminal

Redirection des flux

- >, >> et &> permettent de rediriger le flux de sortie standard et le flux d'erreur standard vers un fichier
- < permet de rediriger le contenu d'un fichier vers l'entrée standard

```
$ ls > fichier.txt
```

- | permet le chaînage de processus, la sortie standard du processus de gauche est redirigée vers l'entrée standard du flux de droite

```
$ cat fichier.txt | grep linux
```

3 / 18

2024-10-14 Les entrées/sorties

Entrée et sorties standards

- Sous UNIX, tout processus possède trois flux orientés caractère :
- Tout d'abord l'entrée standard qui est, par défaut, le clavier.
- Puis la sortie standard qui, par défaut, est l'écran ou le terminal.
- Enfin, la sortie d'erreur standard qui est aussi par défaut l'écran ou le terminal.
- Ces flux peuvent être redirigés.
- Par exemple le symbole > placé après le lancement d'un programme permet de rediriger la sortie standard vers un fichier. Par exemple, la commande `ls > fichier.txt` redirige la liste des fichiers du répertoire courant vers le fichier `fichier.txt`.
- Le symbole >> quant à lui permet de rediriger la sortie d'erreur standard vers un fichier, utile pour capturer et analyser uniquement les messages d'erreur.
- Le symbole &> permet de rediriger à la fois la sortie standard et la sortie d'erreur standard vers un même fichier.

INSA

Rechercher

10 Avril des entrées / sorties

Le langage C // Les entrées / sorties

LE LANGAGE C  
ENTRÉES / SORTIES

- Vocabulaire
- Comment intégrer avec les entrées et sorties standards
- Comment utiliser les fichiers

INSA

Rechercher

10 Avril des entrées / sorties

Le langage C // Les entrées / sorties

Entrée et sorties standards

Trois flux standards sous UNIX orientés caractère

- 1 Entrée standard par défaut le clavier
- 2 Sortie standard par défaut l'écran, le terminal
- 3 Sortie d'erreur standard par défaut l'écran, le terminal

Redirection des flux

- >, >> et &> permettent de rediriger le flux de sortie standard et le flux d'erreur standard vers un fichier

3 / 18

# Remédiation : produire le texte de voix off pour des vidéos (ChatGPT)

Je suis en train de réaliser une vidéo qui présente les entrées sorties en langage C. Je dois rédiger tout ce que je vais dire devant la caméra pour présenter un transparent. Par exemple pour le transparent suivant :

```
\begin{frame}[containsverbatim]
\frametitle{Flux et type de flux}
\begin{block}{Définitions}
\begin{description}
\item[Flux] ensemble de données traitées de manière séquentielle,
par exemple à partir d'un fichier ou d'une connexion réseau
\item[Pointeur de flux] emplacement dans un flux qui indique où
aura lieu le prochain traitement: lecture ou écriture
\item[Type de flux] un flux peut être orienté:
\begin{description}
\item[binaire] les données sont des octets que le programme doit
interpréter
\item[caractère] les données sont des octets représentant des
caractères qui suivent un certain encodage
\end{description}
\end{description}
\end{block}
\end{frame}
```

J'ai décidé de lire le texte suivant:

```
\note{
\begin{itemize}
\item Commençons par définir les concepts de flux, de pointeur de
flux et de type de flux
\item Un flux est un ensemble de données qui est traité de manière
séquentielle.
\item Cela signifie que les données sont lues ou écrites dans un
```

Peux tu rédiger les notes que je dois lire pour présenter le transparent suivant:

```
\begin{frame}[containsverbatim]
\frametitle{Entrée et sorties standards}
\begin{block}{Trois flux standards sous UNIX orientés caractère}
\begin{enumerate}\footnotesize
\item Entrée standard par défaut le clavier
\item Sortie standard par défaut l'écran, le terminal
\item Sortie d'erreur standard par défaut l'écran, le terminal
\end{enumerate}
\end{block}
\begin{block}{Redirection des flux}
\begin{itemize}\footnotesize
\item \texttt{\textbackslash inline[language=bash]{>}, \textbackslash inline[language=bash]{2>}} et
\texttt{\textbackslash inline[language=bash]{&>}} permettent de
rediriger le flux de sortie standard et le flux d'erreur
standard vers un fichier
\item \texttt{\textbackslash inline[language=bash]{<}} permet de rediriger le contenu
d'un fichier vers l'entrée standard
\begin{exampleblock}{}
\begin{lstlisting}[language=bash]
$ ls > fichier.txt
\end{lstlisting}
\end{exampleblock}

\item \texttt{\textbackslash inline[language=bash]{|}} permet le chaînage de
processus, la sortie standard du processus
de gauche est redirigée vers l'entrée standard du flux de
droite
\begin{exampleblock}{}
\end{exampleblock}
```

# Remédiation : produire le texte de voix off pour des vidéos (ChatGPT)

Je veux ce contenu soit plus parlant et rédiger en LaTeX comme je te l'ai précisé (utilisation de la commande \note et une phrase un item)



Mémoire mise à jour

Voici un texte que vous pourriez lire pour accompagner ce transparent :

## 1. Commençons par explorer les flux standards en UNIX.

Sous UNIX, il existe trois flux standards qui sont tous orientés caractère.

- **Le premier est l'entrée standard.** Par défaut, les données proviennent du clavier.
- **Le deuxième est la sortie standard.** Par défaut, les données s'affichent sur l'écran ou dans le terminal.
- **Enfin, nous avons la sortie d'erreur standard,** qui, par défaut également, s'affiche sur l'écran ou le terminal.

## 2. Ces flux peuvent être redirigés pour un traitement plus flexible des données.

- **La redirection vers un fichier :**

Les opérateurs `>` et `2>` permettent de rediriger respectivement la sortie standard et la sortie d'erreur standard vers un fichier. L'opérateur `&>` combine les deux et les redirige simultanément. Par exemple :

```
bash
```

Copier le code



Voici une version rédigée avec la commande `\note` en LaTeX et structurée comme demandé :

```
latex
\begin{document}
\note{
\begin{itemize}
\item Commençons par présenter les trois flux standards sous UNIX.
\item Tout d'abord, il y a l'entrée standard, qui est par défaut le clavier.
\item Ensuite, nous avons la sortie standard, qui affiche les données sur l'écran.
\item Enfin, il y a la sortie d'erreur standard, qui s'affiche également par défaut.
\item Ces flux standards peuvent être redirigés pour manipuler les données de manière flexible.
\item L'opérateur \texttt{>} permet de rediriger la sortie standard vers un fichier.
\item Par exemple, la commande \texttt{ls > fichier.txt} enregistre le résultat de la commande \texttt{ls} dans le fichier \texttt{fichier.txt}.
\item L'opérateur \texttt{2>} permet de rediriger la sortie d'erreur standard vers un fichier.
\item L'opérateur \texttt{\&>} combine la sortie standard et la sortie d'erreur vers un fichier.
\item L'opérateur \texttt{<} permet de rediriger un fichier vers l'entrée standard.
\item Par exemple, on peut utiliser un fichier en tant qu'entrée pour une commande.
\item Enfin, l'opérateur \texttt{|}, appelé pipe, permet de chaîner deux processus.
\item La sortie standard du processus de gauche est envoyée directement en entrée du processus de droite.
\item Par exemple, la commande \texttt{cat fichier.txt | grep linux} utilise le pipe pour filtrer le contenu de \texttt{fichier.txt} en fonction du mot-clé \texttt{linux}.
\item Ces outils rendent la manipulation des flux très puissante et flexible en UNIX.
\end{itemize}
}
```

Copier le code

Ce texte est prêt à être lu lors de votre présentation et suit votre structure préférée.



# Produire des QCM d'auto évaluation (ChatGPT)

LaTeX en 30 min  
où presque...

Nicolas Delestre

LaTeX en 30 min

1 / 28

Quelques exemples 2 / 8



Pour insérer du pseudocode

**procédure** echanger (E/S a,b : Entier)

**Déclaration** tmp : Entier

**debut**

tmp ← a

a ← b

b ← tmp

**fin**

```
1 \procedure(echanger)
2 {\paramEntreeSortie(a,b : \entier)}
3 {tmp: \entier}
4 {\affecter(tmp){a}}
5 {\affecter(a){b}}
6 {\affecter(b){tmp}}
7 }
```



LaTeX en 30 min

5 / 28



LaTeXEn30Min.pdf

PDF

Voici un cours d'introduction au langage LaTeX. Crée dix questions à choix multiples avec à chaque fois trois mauvaises réponses et deux bonnes réponses. L'évaluation de chaque réponse aura une note comprise entre -1 et 1 point. Chaque bonne réponse cochée rapportera 1/2 point. Chaque mauvaise réponse choisie rapportera -1/3 de point. Je veux une précision de 5 chiffres après la virgule pour les nombres flottant. Je veux ces questions au format XML Moodle.

# Produire des QCM d'auto évaluation (ChatGPT)

Voici dix questions à choix multiples au format XML Moodle basées sur un cours d'introduction au langage LaTeX. Chaque question comporte deux bonnes réponses et trois mauvaises réponses, avec une notation précise à cinq chiffres après la virgule. Les bonnes réponses rapportent 0.50000 point chacune et les mauvaises réponses infligent une pénalité de -0.33333 point chacune.

```
xml Copier le code

<quiz>
  <!-- Question 1 -->
  <question type="multichoice">
    <name>
      <text>Qu'est-ce que LaTeX ?</text>
    </name>
    <questiontext format="html">
      <text><![CDATA[LaTeX est :]]></text>
    </questiontext>
    <answer fraction="50.00000">
      <text>Un langage de programmation utilisé pour la composition de documents sc
    </answer>
    <answer fraction="50.00000">
```

# Produire des QCM d'auto évaluation (ChatGPT)

☐ ⚙️ Question ⚙️  
 T Nom de question / Numéro  
 ▲ d'identification

Actions ⚙️ Statut ⚙️

☐ ☰ Comment insère-t-on une  
 équation mathématique en  
 LaTeX ? ✎

Modifier ▾

Prête ▴▾

Pour insérer une équation mathématique :

☐ ☰ Qu'est-ce que LaTeX ? ✎

Modifier ▾

Prête ▴▾

# Trouver des sujets d'examens (ChatGPT)

Huffman, d'endogramme, quadtree, arbre de decision, sont tous des algorithmes qui utilisent des structures d'arbre. Peux tu m'en proposer d'autres ?



Bien sûr ! Voici une liste d'algorithmes et concepts reposant sur des **structures d'arbre** :

## 1. Arbres AVL

- Un type d'arbre binaire de recherche auto-équilibré.
- Utilisé pour garantir un temps de recherche logarithmique.

## 2. Arbres Red-Black (Rouge-Noir)

- Un autre type d'arbre auto-équilibré.
- Principalement utilisé dans des bibliothèques standard (par exemple, les ensembles et les cartes en C++/Java).

## 3. B-Arbres

- Utilisés dans les bases de données et les systèmes de fichiers pour gérer efficacement de grandes quantités de données sur disque.

## 4. Arbres Trie (ou Patricia)

# Trouver des sujets d'examens (ChatGPT)

Peux tu présenter plus en détail les arbres de fenwick



Les **arbres de Fenwick**, également appelés **Binary Indexed Trees (BIT)**, sont une structure de données efficace pour manipuler des **séries numériques**, en particulier lorsqu'on a besoin de calculer rapidement des **sommes cumulées** et d'effectuer des **mise à jour dynamiques**. Voici une présentation détaillée :

## 1. Fonctionnalités principales :

- **Calcul des sommes cumulées** dans un tableau (de l'indice 1 à  $i$ ) en  $O(\log n)$ .
- **Mise à jour d'un élément** ou d'une plage dans le tableau en  $O(\log n)$ .

Ces opérations sont beaucoup plus rapides qu'avec une implémentation naïve, où les sommes cumulées ou les mises à jour nécessitent  $O(n)$ .

# Rédigez des sujets et corrections d'examens (ChatGPT)

o

Voici un exemple de TAD exprimés en LaTeX spécifiant les cluster hiérarchiques.

```
\begin{tad}
\tadNom{ClusteringHierarchique}
\tadParametres{Donnee}
\tadDependances{\reelPositif, \booleen}
\begin{tadOperations}{obtenirFilsGauche}
\tadOperation{feuille}{Donnee}{ClusteringHierarchique}
\tadOperation{noeud}{\tadTroisParams{\reelPositif}
{ClusteringHierarchique}{ClusteringHierarchique}}
{ClusteringHierarchique}
\tadOperation{estUneFeuille}{ClusteringHierarchique}{\booleen}
\tadOperationAvecPreconditions{obtenirDonnee}
{ClusteringHierarchique}{Donnee}
\tadOperationAvecPreconditions{obtenirFilsGauche}
{ClusteringHierarchique}{ClusteringHierarchique}
\tadOperationAvecPreconditions{obtenirFilsDroit}
{ClusteringHierarchique}{ClusteringHierarchique}
\tadOperationAvecPreconditions{obtenirDistance}
```

```
\end{tadPreconditions}
\begin{tadAxiomes}
\tadAxiome{estUneFeuille(feuille(d))}
\tadAxiome{non estUneFeuille(noeud(v,chg, chd))}
\tadAxiome{obtenirDonnee(feuille(d))=d}
\tadAxiome{obtenirFilsGauche(noeud(v,chg, chd))=chg}
\tadAxiome{obtenirFilsDroit(noeud(v,chg, chd))=chd}
\tadAxiome{obtenirDistance(noeud(v,chg, chd))=v}
\end{tadAxiomes}
\end{tad}
```

Peux tu spécifier le TAD ArbreFenwick sachant que les opérations sont :

créer un arbre de Fenwick à partir d'un nombre de valeurs nulles;  
 savoir si un arbre de Fenwick est une feuille;  
 obtenir la valeur lorsque l'arbre de Fenwick est une feuille;  
 obtenir les deux arbres de Fenwick (fils gauche et droit) lorsque l'arbre de Fenwick n'est pas une feuille;  
 obtenir les indices d'intervalle lorsque le clustering hiérarchique n'est pas une feuille;  
 obtenir la somme des valeurs sur l'intervalle lorsque le clustering

# Rédigez des sujets et corrections d'examens (ChatGPT)

Voici une spécification en LaTeX du TAD pour l'**Arbre de Fenwick** en suivant votre style et vos conventions :

latex

 Copier le code

```
\begin{tad}
  \tadNom{ArbreFenwick}
  \tadParametres{Valeur}
  \tadDependances{\naturel, \intervalle, \boolean}

  \begin{tadOperations}{obtenirFilsGauche}
    \tadOperation{creer}{\naturel}{ArbreFenwick}
    \tadOperation{feuille}{\naturel}{ArbreFenwick}
    \tadOperation{noeud}{\tadQuatreParams{\intervalle}{\naturel}{ArbreFenwick}{ArbreFenwick}}
    \tadOperation{estUneFeuille}{ArbreFenwick}{\boolean}
    \tadOperationAvecPreconditions{obtenirValeur}{ArbreFenwick}{\naturel}
    \tadOperationAvecPreconditions{obtenirFilsGauche}{ArbreFenwick}{ArbreFenwick}
    \tadOperationAvecPreconditions{obtenirFilsDroit}{ArbreFenwick}{ArbreFenwick}
    \tadOperationAvecPreconditions{obtenirIntervalle}{ArbreFenwick}{\intervalle}
    \tadOperationAvecPreconditions{obtenirSomme}{ArbreFenwick}{\naturel}
    \tadOperationAvecPreconditions{incrementer}{\tadDeuxParams{ArbreFenwick}{\tadDeuxParams}}
  \end{tadOperations}

  \begin{tadPreconditions}{obtenirFilsGauche(af)}
    \tadPrecondition{obtenirValeur(af)}{estUneFeuille(af)}
    \tadPrecondition{obtenirFilsGauche(af)}{non estUneFeuille(af)}
    \tadPrecondition{obtenirFilsDroit(af)}{non estUneFeuille(af)}
    \tadPrecondition{obtenirIntervalle(af)}{non estUneFeuille(af)}
    \tadPrecondition{obtenirSomme(af)}{non estUneFeuille(af)}
  \end{tadPreconditions}
```

# Rédigez des sujets et corrections d'examens (ChatGPT)

On passe à la conception préliminaire. Voici les signatures des fonctions ou procédure pour la TAD ClusterHierarchique :

```
\signatureFonction{feuille}{d: Donnee}{ClusteringHierarchique}{}
```

```
\signatureFonction{noeud}{v: \reel, chg, chd:
```

```
ClusteringHierarchique}{ClusteringHierarchique}{}
```

```
\signatureFonction{estUneFeuille}{ch:
```

```
ClusteringHierarchique}{\booleen}{}
```

```
\signatureFonction{obtenirDonnee}{ch: ClusteringHierarchique}
```

```
{Donnee}{estUneFeuille(ch)}}
```

```
\signatureFonction{obtenirDistance}{ch:
```

```
ClusteringHierarchique}{\reel}{non estUneFeuille(ch)}}
```

```
\signatureFonction{obtenirFilsGauche}{ch:
```

```
ClusteringHierarchique}{\reel}{non estUneFeuille(ch)}}
```

```
\signatureFonction{obtenirFilsDroit}{ch:
```

```
ClusteringHierarchique}{\reel}{non estUneFeuille(ch)}}
```


Peux tu me donner celles pour la TAD ArbreFenwick



# Rédigez des sujets et corrections d'examens (ChatGPT)

Voici les signatures des fonctions et procédures pour la TAD **ArbreFenwick**, en suivant le même style et vos contraintes :

latex

 Copier le code

```

\signatureFonction{creer}{n: \naturel}{ArbreFenwick}{}
\signatureFonction{feuille}{v: \naturel}{ArbreFenwick}{}
\signatureFonction{noeud}{i: \intervalle, s: \naturel, afg, afd: ArbreFenwick}{ArbreFenwick}{}
\signatureFonction{estUneFeuille}{af: ArbreFenwick}{\booleen}{}
\signatureFonction{obtenirValeur}{af: ArbreFenwick}{\naturel}{estUneFeuille(af)}
\signatureFonction{obtenirFilsGauche}{af: ArbreFenwick}{ArbreFenwick}{non estUneFeuille(af)}
\signatureFonction{obtenirFilsDroit}{af: ArbreFenwick}{ArbreFenwick}{non estUneFeuille(af)}
\signatureFonction{obtenirIntervalle}{af: ArbreFenwick}{\intervalle}{non estUneFeuille(af)}
\signatureFonction{obtenirSomme}{af: ArbreFenwick}{\naturel}{non estUneFeuille(af)}
\signatureProcEDURE{incrementer}{af: \tadVar{ArbreFenwick}, ind, delta: \naturel}{}

```

# Rédigez des sujets et corrections d'examens (Copilot)

```

270 \subsubsection{Algorithme de \texttt{estPresent}}
271 \begin{enumerate}
272 \item Expliquez comment un algorithme pourrait indiquer que le mot \textit{ces} est
273 présent dans le dictionnaire de la figure~\ref{fig:dico1};
274 \item Expliquez comment un algorithme pourrait indiquer que le mot \textit{da}
275 n'est pas présent dans le dictionnaire la figure~\ref{fig:dico1};
276 \item Expliquez comment un algorithme pourrait indiquer que le mot
277 \textit{et} n'est pas présena dans le dictionnaire de la
278 figure~\ref{fig:dico1}.
279 \item Donnez l'algorithme récursif de \texttt{estPresent}.
280 \end{enumerate}
281
282 \begin{solution}
283 \begin{algorithm}
284 \function{estPresent}{d: Dictionnaire, m: Mot}{\boolean}
285 {
286   \retourner{estPresentRec(d,m,1)}
287 }
288 }

```

```

}
\function{estPresentRec}{d: Dictionnaire, m: Mot, i: \natnum}
{
{
\si{longueur(m)=i}
{
\retourner{estPresentFinDeMot(d)}
}
\si{estVide(d)}
{
\retourner{faux}
}
\si{plusPetit(iemeLettre(m,i),racine(d).lettre)}
{
\retourner{estPresentRec(filsGauche(d),m,i+1)}
}
\si{plusPetit(racine(d).lettre,iemeLettre(m,i))}
{
\retourner{estPresentRec(filsDroit(d),m,i)}
}
}
}

```

# Rédigez des sujets et correction d'examens : résultat

- si le nœud courant est une feuille, la somme préfixe vaut  $v_i$ .  
Soit les 9 valeurs  $v_i$  suivantes :

2	1	3	1	4	5	8	2	1
1	2	3	4	5	6	7	8	9

- Dessinez l'arbre de Fenwick pour ces valeurs. Vous pouvez à indiquer pour les nœuds non feuille les intervalles et les sommes des valeurs sur ces intervalles. Pour les feuilles vous indiquerez  $i$  et  $v_i$ .
- En utilisant l'arbre que vous venez de dessiner, calculez la somme préfixes pour l'indice 7. Expliquez votre démarche.

## 2.2 Analyse (4 compétences)

On considère que les valeurs stockées sont de type entier et qu'elles sont indexées à partir de 1.

Soit le TAD `ArbreFenwick` pour lequel on peut :

- créer un arbre de Fenwick à partir d'un nombre de valeurs, toutes nulles ;
  - savoir si un arbre de Fenwick est une feuille ;
  - obtenir les deux arbres de Fenwick (fils gauche et droit) lorsque l'arbre de Fenwick n'est pas une feuille ;
  - obtenir l'indice d'une valeur lorsque l'arbre de Fenwick est une feuille ;
  - obtenir les indices de l'intervalle lorsque l'arbre de Fenwick n'est pas une feuille ;
  - obtenir la valeur se trouvant à la racine d'un arbre de Fenwick :  $v_i$  si c'est une feuille,  $\sum_{k=i}^f v_k$  sinon
  - ajouter un entier à une valeur identifiée à partir de son indice.
- Dérivez ce TAD (sans les axiomes et sémantiques).

## 2.3 Conception préliminaire (2 compétences)

Proposez les signatures des fonctions et procédures correspondant aux opérations du TAD `ArbreFenwick`.

## 2.4 Conception détaillée (7 compétences)

On décide de représenter le type `ArbreFenwick` à l'aide de la SDD `ArbreBinaire` (cette SDD est appelée en Annexe) de la façon suivante :

**Type Contenu = Structure**

**val : Entier**

**iDebut : NaturelNonNul**

**iFin : NaturelNonNul**

**fStructure**

**Type ArbreFenwick = ArbreBinaire <Contenu>**

Tel que :

- `val` est la valeur si le nœud courant est une feuille, ou si ce n'est pas une feuille, `val` est égale à la somme des valeurs contenu dans ses deux sous arbres ;
- `iDebut`, `iFin` sont les indices de l'intervalle, on l'écrit `iDebut = iFin` pour les feuilles.
- On rappelle que l'arbre de Fenwick permet de partir à jour efficacement une des valeurs  $v_i$ . Cette mise à jour doit impacter les éléments concernés de l'arbre. Donnez l'algorithme itératif de la fonction ou de la procédure qui permet d'ajouter un entier à une valeur désignée par son indice (opération numéro 8 de la question 2.2)

## 2.5 Utilisation : somme préfixes (9 compétences)

Donnez l'algorithme récursif de la fonction ou de la procédure qui permet de calculer la somme préfixes des  $i$  premières valeurs d'un arbre de Fenwick. Si  $i$  est plus grand que le nombre de valeurs alors la somme préfixes sera la somme de toutes ces valeurs.

Votre algorithme récursif est-il terminal ? Qu'est-ce que cela implique ? Justifiez

### Attendus d'apprentissages disciplinaires évalués

- AN201 : Identifier les dépendances d'un TAD
- AN203 : Savoir si une opération identifiée fait partie du TAD à spécifier
- AN204 : Formaliser des opérations d'un TAD
- AN205 : Formaliser les préconditions d'une opération d'un TAD

**Nom :** `ArbreFenwick`

**Utilise :** `Boolean`, `NaturelNonNul`, `Entier`, `Liste`

**Opérations :** `aAPartirDeNZero : NaturelNonNul → ArbreFenwick`

`aAPartirDeValeurs : Liste <Entier> → ArbreFenwick`

`estUneFeuille : ArbreFenwick → Boolean`

`obtenirFilsGauche : ArbreFenwick → ArbreFenwick`

`obtenirFilsDroit : ArbreFenwick → ArbreFenwick`

`obtenirIndice : ArbreFenwick → NaturelNonNul`

`obtenirIntervalle : ArbreFenwick → NaturelNonNul × NaturelNonNul`

`obtenirValeur : ArbreFenwick → Entier`

`ajouter : ArbreFenwick × NaturelNonNul × Entier → ArbreFenwick`

**Préconditions :** `aAPartirDeValeurs()`: non estVide()

`obtenirFilsGauche(ad)`: non estUneFeuille(ad)

`obtenirFilsDroit(ad)`: non estUneFeuille(ad)

`obtenirIndice(a)`: non estUneFeuille(a)

`obtenirIntervalle(ad)`: non estUneFeuille(ad)

`ajouter(ad,i,v)`: (estUneFeuille(ad) et obtenirIndice(ad)=i) ou (non estUneFeuille(ad) et avec d,f ← obtenirIntervalle(ad), i ∈ d..f)

## 2.3 Conception préliminaire (2 compétences)

Proposez les signatures des fonctions et procédures correspondant aux opérations du TAD `ArbreFenwick`.

**Solution proposée :**

### Attendus d'apprentissages disciplinaires évalués

- CP003 : Choisir entre une fonction et une procédure
- CP004 : Conserver une signature (préconditions incluses)

— fonction `aAPartirDeNZero (n : NaturelNonNul) : ArbreFenwick`

— fonction `aAPartirDeValeurs (vs : Liste <Entier>) : ArbreFenwick`

[précondition(a) non estVide()]

— fonction `estUneFeuille (af : ArbreFenwick) : Boolean`

— fonction `obtenirFilsGauche (af : ArbreFenwick) : ArbreFenwick`

[précondition(a) non estUneFeuille(ad)]

— fonction `obtenirFilsDroit (af : ArbreFenwick) : ArbreFenwick`

[précondition(a) non estUneFeuille(ad)]

— fonction `obtenirIndice (af : ArbreFenwick) : NaturelNonNul`

[précondition(a) estUneFeuille(ad)]

— fonction `obtenirIntervalle (af : ArbreFenwick) : NaturelNonNul × NaturelNonNul`

[précondition(a) non estUneFeuille(ad)]

— fonction `obtenirValeur (af : ArbreFenwick) : Entier`

— procédure `ajouter (E/S af : ArbreFenwick, E i : NaturelNonNul, v : Entier)`

[précondition(a) (estUneFeuille(ad) et obtenirIndice(ad)=i) ou (non estUneFeuille(ad) et avec d,f ← obtenirIntervalle(ad), i ∈ d..f)]

# Analyse bibliographique (NotebookML)

## A SURVEY OF PROMPT ENGINEERING METHODS IN LARGE LANGUAGE MODELS FOR DIFFERENT NLP TASKS

**Shubham Vatsal & Harsh Dubey**  
 Department of Computer Science  
 New York University, CIMS  
 New York, USA  
 {sv2128, hd2225}@nyu.edu

### ABSTRACT

Large language models (LLMs) have shown remarkable performance on many different Natural Language Processing (NLP) tasks. Prompt engineering plays a key role in adding more to the already existing abilities of LLMs to achieve significant performance gains on various NLP tasks. Prompt engineering requires composing natural language instructions called prompts to elicit knowledge from LLMs in a structured way. Unlike previous state-of-the-art (SoTA) models, prompt engineering does not require extensive parameter re-training or fine-tuning based on the given NLP task and thus solely operates on the embedded knowledge of LLMs. Additionally, LLM enthusiasts can intelligently extract LLMs' knowledge through a basic natural language conversational exchange or prompt engineering, allowing more and more people even without deep mathematical machine learning background to experiment with LLMs. With prompt engineering gaining popularity in the last two years, researchers have come up with numerous engineering techniques around designing prompts to improve accuracy of information extraction from the LLMs. In this paper, we summarize different prompting techniques and club them together based on different NLP tasks that they have been used for. We further granularly highlight the performance of these prompting strategies on various datasets belonging to that NLP task, talk about the corresponding LLMs used, present a taxonomy diagram and discuss the possible SoTA for specific datasets. In total, we read and present a survey of 44 research papers which talk about 39 different prompting methods on 29 different NLP tasks of which most of them have been published in the last two years.

39 pages  
 39 techniques de prompt  
 présentées  
 29 tâches de TAL présentées

# Analyse bibliographique (NotebookML)

≡

NotebookLM

Sources

Sélectionner toutes les sources

✓

2005.11401v4.pdf

✓

2201.11903v6.pdf

✓

2203.11171v4.pdf

✓

2205.10625v3.pdf

✓

2205.11822v2.pdf

✓

2210.00720v2.pdf

✓

2210.02406v2.pdf

✓

2210.03493v1.pdf

✓

2210.03629v3.pdf

✓

2211.10435v2.pdf

✓

2211.12588v4.pdf

✓

Prompt LLM

Paramètres

Partager

Enregistrer dans la note

catégorise les différentes techniques de prompt engineering

Catégorisation des Techniques de Prompt Engineering

Les sources fournies mettent en évidence une variété de techniques de prompt engineering qui peuvent être classées en fonction de leur approche et de leur objectif.

**1. Techniques Basées sur le Raisonnement Explicite:**

- **Chain-of-Thought (CoT)** : Encourage le modèle de langage à générer des étapes de raisonnement intermédiaires avant de produire la réponse finale. Cette approche vise à améliorer la performance sur des tâches de raisonnement complexes.
 

1

2

3

4

...

  - Différentes variantes de CoT existent :
    - **Zero-Shot-CoT** : Ne nécessite pas d'exemples et s'appuie sur la capacité du modèle à générer des raisonnements de manière autonome. 

7 11 12 31 ...
    - **Manual-CoT** : Utilise des exemples annotés manuellement pour guider le modèle dans la génération de raisonnements. 

12 31
    - **Auto-CoT** : Génère automatiquement des exemples de raisonnement en regroupant les questions du jeu de données. 

11 12 13 31 34
    - **Self-Consistency (SC)** : Améliore CoT en échantillonnant plusieurs chemins de raisonnement et en sélectionnant la réponse la plus cohérente. 

5 7 10 11 ...
  - **Least-to-Most Prompting (LtM)** : Décompose un problème complexe en sous-problèmes plus simples, résolus séquentiellement par le modèle. Cette technique est efficace pour les tâches impliquant des listes ou des séquences. 

5 7
  - **Program-Aided Reasoning** : Utilise le modèle de langage pour générer du code qui résout le problème. Cette approche tire parti de la capacité des modèles pré-entraînés sur du code, comme Codex, à raisonner logiquement. 












7 13 23 24 42

**INSA**  
 ROUEN NORMANDIE

11 / 14

# Analyse bibliographique (NotebookML)

## NotebookLM

Sources Sélectionner toutes les sources ☒ 2005.11401v4.pdf ☒ 2201.11903v6.pdf ☒ 2203.1171v4.pdf ☒ 2205.10625v3.pdf ☒ 2205.11822v2.pdf ☒ 2210.00720v2.pdf ☒ 2210.02406v2.pdf ☒ 2210.03493v1.pdf ☒ 2210.03629v3.pdf ☒ 2211.10435v2.pdf ☒ 2211.12588v4.pdf ☒

## Prompt LLM

 Paramètres Partager

Qu'est qui différencie le CoT du LtM ?

### Différences entre CoT et LtM

Les deux techniques de prompt engineering, Chain-of-Thought (CoT) et Least-to-Most Prompting (LtM), visent à améliorer les capacités de raisonnement des modèles de langage. Cependant, elles diffèrent dans leur approche de décomposition des problèmes et dans les types de tâches pour lesquelles elles sont les plus efficaces.

#### Chain-of-Thought (CoT):

- **Principe:** Le CoT encourage le modèle à générer des étapes de raisonnement intermédiaires explicites avant de produire la réponse finale <sup>1 2 3 4</sup>. Cette approche imite le processus de pensée humain en décomposant un problème complexe en étapes logiques plus simples.
- **Fonctionnement:** Une invite telle que "Réfléchissons étape par étape" est ajoutée à la question pour inciter le modèle à articuler son raisonnement <sup>2 5</sup>.
- **Avantages:**
  - ☐ Améliore la performance sur des tâches de raisonnement complexes en rendant le processus de raisonnement plus transparent.
  - ☐ Permet d'identifier les erreurs de raisonnement du modèle en examinant les étapes intermédiaires.
- **Inconvénients:**
  - ☐ Peut être moins efficace pour les tâches simples ou pour les modèles de langage moins puissants.
  - ☐ La qualité du raisonnement généré dépend de la capacité du modèle à comprendre l'invite "Réfléchissons étape par étape".
- **Variantes:**
  - ☐ Zero-Shot-CoT <sup>2</sup> : Aucun exemple de raisonnement n'est fourni au modèle.
  - ☐ Manual-CoT <sup>2</sup> : Des exemples annotés manuellement sont utilisés pour guider le modèle.
  - ☐ Auto-CoT <sup>1</sup> : Des exemples de raisonnement sont générés automatiquement.
  - ☐ Self-Consistency (SC) <sup>6 7</sup> : Plusieurs chemins de raisonnement sont échantillonnés, et la réponse la plus cohérente

$\Rightarrow$ **INSA**  
ROUEN NORMANDIE

# Reformater des données (ChatGPT)



ReferentielAppTIIA-Version-JUIN-20...

PDF

je dois créer un XML à partir d'un tableau PDF.

Par exemple ce PDF produit le XML :

```
<savoir-agir id="specifier">
```

```
  <titre xml:lang="fr">Spécifier</titre>
```

```
  <titre xml:lang="en">Specify</titre>
```

```
  <description xml:lang="fr">
```

Formaliser un système d'IA en produisant des documents identifiants les entrées, les sorties et les métriques permettant d'évaluer la performance du système.

```
  </description>
```

```
  <description xml:lang="en">
```

Formalize an AI system by producing documents identifying the inputs, outputs and metrics to evaluate the system's performance.

```
  </description>
```

```
  <cadre xml:lang="fr">
```

En utilisant des langages et formalismes reconnus ainsi qu'en respectant les directives de protection des données, d'accessibilité et de développement durable et responsabilité sociétale



# Reformater des données (ChatGPT)



ReferentielAppTIIA-Version-JUIN-20...

PDF

Peux tu me générer le XML associé à ce nouveau PDF



Based on the information extracted from the PDF, here is the XML for the "Modéliser" competence from your document:

xml

Copier le code

```
<savoir-agir id="modeliser">
  <titre xml:lang="fr">Modéliser</titre>
  <titre xml:lang="en">Model</titre>
  <description xml:lang="fr">
    Identifier la classe de modèle adaptée au problème posé. Pour cela il ident
  </description>
  <description xml:lang="en">
    Identify the appropriate model class for the problem posed. This involves i
  </description>
<cadre xml:lang="fr">
```

# Conclusion

## J'utilise les LLM...

- pour trouver des idées
- pour gagner du temps
- pour corriger ce que je rédige
- pour réaliser des tâches répétitives

## Techniques de prompt

- Poser simplement la question de manière détaillée, en prenant quelques fois un rôle, suffit pour avoir de « bonnes » réponses
- Dans certains contextes, la technique du *One-shot* est utile

# Si c'est gratuit, c'est toi le produit...

D'après ce que tu sais de moi, dessine une image qui me représente



Pour te représenter, voici ce que j'imagine pour l'image :

Je vois une scène qui évoque l'ingénierie pédagogique dans un contexte universitaire avec des éléments techniques et académiques. Dans cette scène, une personne se tient devant un tableau ou un écran de présentation (conçu en LaTeX, avec le style de Beamer), expliquant des concepts en ingénierie. L'arrière-plan inclurait des détails subtils tels qu'un diagramme ou des éléments de code en C et Python. Quelques symboles mathématiques ou composants techniques (tableaux, spécifications en LaTeX) renforcent l'atmosphère académique et technique.

Je vais maintenant générer cette image !

