* Deux model qui se bagarre : Time based sharing model TBRS
* Modèle de la mémoire de travail à la frontière entre psycho et math
  + Intégré des math = prédiction

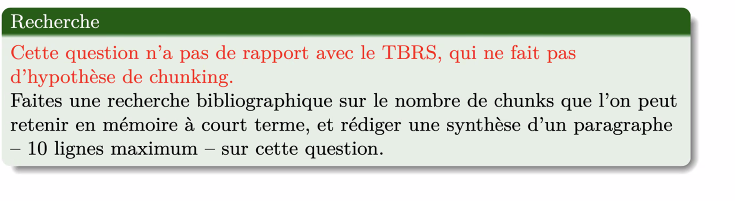
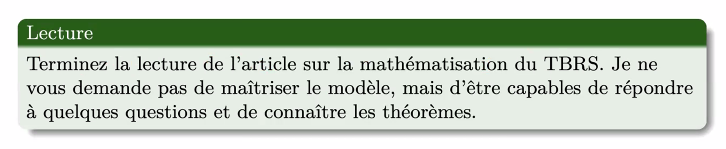
Principes du TBRS :

* Comment les gens font pour rappeler les lettres en mémoire de travail avec une tâche intermittente secondaire ?
  + 3 évents : présentation de lettre, tache concurrente/secondaire, temps libre
  + Effet : lettrer → temps → tache : rappel ++ // lettre → tache → temps : rappel –
  + Ici on prend en compte le temps suffit à expliquer l’effacement en mémoire
* Principe de base du TBRS :
  + A chaque instant l’attention est focalisée soit sur **une** lettre à rappeler (un item), soit sur la tâche secondaire
  + Lorsque l’attention est focalisée sur un item, l’**activation** de l’item augmente. Sinon, elle décroit
  + Cette loi de croissance ou de décroissance ne dépend pas des autres items ou du temps en soi, mais seulement de l’activation actuelle
    - Temps : même augmentation au début et à la fin de l’expérience (pas d’effet de fatigue ou quoi)
    - De l’item : same pour chaque item
    - Mais la façon dont ça monte ou ça descend peut-être choisi, une exp, un logarithmique
  + La probabilité de rappel correct d’un item à un moment donnée est une fonction croissante de l’activation
    - Plus l’item est activé, plus y’a de chance de le rappeler
  + Le **cognitive load** est la proportion du temps passée à ne pas se focaliser sur un item. L’activation dépend du cognitive load et du temps total, mais pas de la répartition des moment passés à réactiver
    - Exemple : lettre → 10s → tache → rappel <=> lettre → 3s → tache → 7s → rappel
* L’article :
  + Complex span task : présentation d’item + tache secondaire à effectuer
  + 3 taches concurrentes utilisé : lire des lettres, vérifier des calculs mathématiques, update un chiffre en mémoire, répétition de syllabe.
  + Cognitive load : la proportion du temps passée à ne **pas** se focaliser sur un item.
  + Critiques TBRS\* : Connexionniste
    - S’éloigne du TBRS original en intégrant d’autre modèle
    - Intégration non nécessaire de certaine chose (eg. Serial position coding)
      * Choix de l’implémentation de comment coder les lettres dans le modèle
      * Rajout d’hypothèse par rapport au TBRS
    - Modèle connexionniste = boite noire, comme en deep learning, on a juste des poids de neurone
    - Beaucoup de paramètre en plus par rapport au TBRS originel. Moins de paramètre est préférable. Sinon on modélise tous les cas possibles etc.
* Démontrer que si les HP du TBRS sont vérifiées (strictement), alors il n’y a jamais d’oubli possible
  + HP du cognitive load
  + Exemple :
    - A → tache
    - → baisse de l’activation : jusqu’avant l’oublie
    - → temps libre : jusqu’à qu’on retourne au niveau de base
    - → boucle : ainsi de suite
    - On n’oublie pas
  + Est-ce qu’en 30 s
  + Pas compris pourquoi
  + Mais apparemment c’étais vraiment comme la demi-vie

Probabilité, chances et log-odds

* Odds = les chances,
* Valeur possible :
* Multiplier la chance par deux ne revient pas à multiplier la proba par deux

Fonction de tache, de focus et d’activation

* Lecture de la part Overview de l’artcile
  + Comment est définie l’activation ? par la log-odds
  + Q2
  + Q3 : Dépend de la durée du focus, définis l’alternance entre les deux ou n lettre. Combien de temps on reste focus sur un item avant de passer au suivant, fixe ou variable
  + Q4 : pas necessairement dans l’autre
  + Q5 : dépends des taux de rafraichissement et ; a quel vitesse sa croit/décroit
* Autre stratégie possible
* ……
* Faire un petit pragraphe sur le nombre de chunk en mémoire DEMANDER A JOZE
* Lire l’arcile et avoir l’idée de comment ça fonctionne, comment ça a été mathématisé

Devoir :

Une chose est sûre, c’est que la mémoire à court terme à une capacité limité. On mesure généralement cette tâche par une tache de mémorisation-restitution. George Miller a suggéré en 1956 que la mémoire de travail avait une capacité de 7+-2, et cela s’accordé bien avec les découverte et recherche de l’époque. De nos jour, ce chiffre est à nuancer. En effet il correspond bien à la capacité des étudiants (participant majoritaire des études de psychologie) à retenir des suite de chiffre. Mais le matériel et la population testé exercent une influence sur le résultats. En effet des mots phonologiquement proche ou des mots long à prononcer sont plus difficile à restituer.   
Un chiffre plus précis aujourd’hui est de 4 chunks ou pièce d’information. Mais certaine théorie argumente contre une mesure pas un nombre d’éléments.

Article :

New Cours du 22/02/2022 :

Machine de Turing : on a fait plein de truc

La complexité algorithmique d’une chaine de caractère c’est la longueur du plus court programme qui permet de la mettre sur une machine de Turing universelle.

Démontrer que : Si alors

Exo 2 : où ne dépend que de T

Exo 3 :  : k représente le programme qui « émule » dans l’autre machine (exemple mac vs windows)

Probabilité algorithmique : on prend un programme au hasard et on regarde les chance qu’il nous donne une suite donnée fixé choisis

Coding théorème

Problème de la machine de Turing :

* Ça dépend de la machine universelle choisis
* Ce n’est pas calculable, pas d’algo pour le trouver

Il existe un packet pour estimer ça : ACSS. Il a expliqué mais bon

* Je crois on fait tourner plein plein de machine de turing aléatoire et on regarde en moyenne ce que ca donne

L’entorpie (compression) ne dépend uniquement du nombre de 0 et de 1 de la suite

* 011001 -> entropie max
* // 000000 -> Entropie minimale
* 010101 -> entropie same que le premier alors que d’un point de vu complexité d’algo c’est instinctivement plus simple