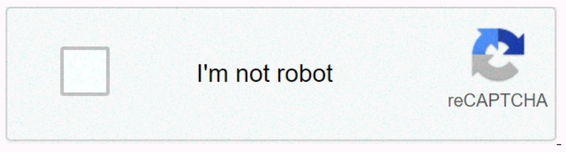
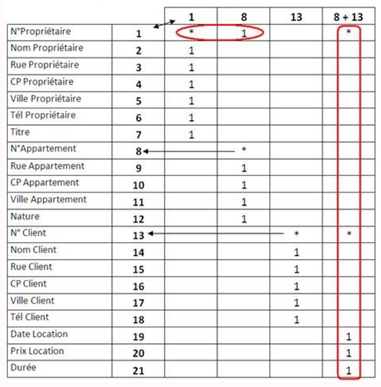


Continue

Algorithme matrice exercice corrige pdf

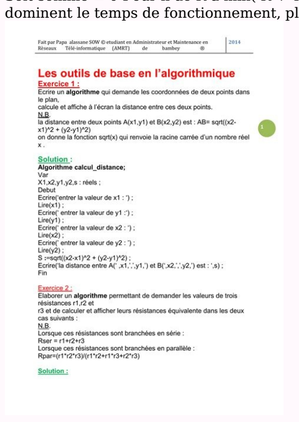
Contenu de sensagent définitionssynonymesantonymesencyclopédie dictionnaire et traducteur pour sites web Alexandria Une fenêtre (pop-into) d'information (contenu principal de Sensagent) est invoquée un double-clic sur n'importe quel mot de votre page web. LA fenêtre fournit des explications et des traductions contextuelles, c'est-à-dire sans obliger votre visiteur à quitter votre page web ! Essayer ici, télécharger le code; SensagentBox Avec la boîte de recherches Sensagent, les visiteurs de votre site peuvent également accéder à une information de référence pertinente parmi plus de 5 millions de pages web indexées sur Sensagent.com. Vous pouvez Choisir la taille qui convient le mieux à votre site et adapter la charte graphique. Solution commerce électronique Augmenter le contenu de votre site Ajouter de nouveaux contenus Add à votre site depuis Sensagent par XML.

Parcourir les produits et les annonces Obtenir des informations en XML pour filtrer le meilleur contenu. Indexer des images et définir des méta-données Fixer la signification de chaque méta-donnée (multilingue). Renseignements suite à un email de description de votre projet. Lettris Lettris est un jeu de lettres gravitationnelles proche de Tetris.



		1	8	13	8 + 13
N°Propriétaire	1	*	1		*
Nom Propriétaire	2	1			
Rue Propriétaire	3	1			
CP Propriétaire	4	1			
Ville Propriétaire	5	1			
Tél Propriétaire	6	1			
Titre	7	1			
N°Appartement	8		*		
Rue Appartement	9		1		
CP Appartement	10		1		
Ville Appartement	11		1		
Nature	12		1		
N° Client	13			*	*
Nom Client	14			1	
Rue Client	15			1	
CP Client	16			1	
Ville Client	17			1	
Tél Client	18			1	
Date Location	19				1
Prix Location	20				1
Durée	21				1

Chaque lettre qui apparaît descend ; il faut placer les lettres de telle manière que des mots se forment (gauche, droit, haut et bas) et que de la place soit libérée. boggle Il s'agit en 3 minutes de trouver le plus grand nombre de mots possibles de trois lettres et plus dans une grille de 16 lettres. Il est aussi possible de jouer avec la grille de 25 cases. Les lettres doivent être adjacentes et les mots les plus longs sont les meilleurs. Participer au concours et enregistrer votre nom dans la liste de meilleurs joueurs ! Jouer Dictionnaire de la langue françaisePrincipales Références La plupart des définitions du français sont proposées par SenseGates et comportent un approfondissement avec Littré et plusieurs auteurs techniques spécialisés. Le dictionnaire des synonymes est surtout dérivé du dictionnaire intégral (TID). L'encyclopédie française bénéficie de la licence Wikipedia (GNU). Traduction Changer la langue cible pour obtenir des traductions. Astuce: parcourir les champs sémantiques du dictionnaire analogique en plusieurs langues pour mieux apprendre avec sensagent. 4623 visiteurs en ligne calculé en 0,047s Page 2 Page 3 Page 4 Page 5 Complexité asymptotique - notations Les notations asymptotiques sont des outils mathématiques permettant de représenter la complexité temporelle des algorithmes d'analyse asymptotique. Les 3 notations asymptotiques suivantes...
Lire la suite Page 6 Chaines de caractères en Python Les chaines de caractères sont des listes de caractères. On parle de chainage car les caractères se suive et chaque caractère a sa place comme les maillons d'une chaine. Il est ainsi... Lire la suite Ecole préparatoire en Sciences & Techniques d'Oran epst-oran Intitulé : Informatique 1 ère année, Semestre 2, 2011/2012 analyse numérique exercices corrigés sur les tableaux et les matrices extrait de pdf: Exercice1 : Ecrire un algorithme en utilisant une fonction pour sommer des éléments d'un tableau d'entiers.Exercice2 : Ecrire deux fonctions : une de calcul de moyenne d'un tableau et l'autre pour extraire le minimum des éléments dans un tableau.Exercice3 : Ecrire une fonction qui calcule le nombre d'occurrences d'un élément donné dans un tableau.Exercice 4 : Ecrire l'algorithme qui saisit deux matrices A et B (2,3) par des nombres réels, calcule la somme suivante C = 2*A-3*B puis affiche C.Exercice 5 : Ecrire sous forme d'une procédure la somme de deux matrices réellesExercice 6 :Soit une matrice carrée. Ecrire l'algorithme qui permet de faire la somme de la diagonale principale de cette matrice ...Exercice 7 : Ecrire un algorithme qui permet de : - Saisir une matrice T(2,4) d'entiers. - Calculer P le nombre des éléments pairs. - Calculer R le nombre des éléments impairs. - Afficher P et R.Exercice 8 :Soit une matrice M de 200 lignes et 100 colonnes à valeurs entières . donner une fonction qui détermine la ligne dont la somme ddes elements est maximale Nom du fichier : analyse numérique exercices corrigées les tableaux + les matrices By ExoSup.pdf Taille du fichier : 416 KB Date de publication : 06/09/2015 Télécharger ICI-ICI-ICI-ICI-ICI Analyse Numérique et Algorithme,Analyse Numérique et Algorithme td,smp,smp s3 Parce que la multiplication matricielle est une opération si centrale dans de nombreux algorithmes numériques , beaucoup de travail a été investi pour rendre les algorithmes de multiplication matricielle efficaces. Les applications de la multiplication matricielle dans les problèmes informatiques se trouvent dans de nombreux domaines, y compris le calcul scientifique et la reconnaissance de formes, ainsi que dans des problèmes apparemment sans rapport tels que le comptage des chemins à travers un graphique . De nombreux algorithmes différents ont été conçus pour multiplier des matrices sur différents types de matériel, y compris des systèmes parallèles et distribués , où le travail de calcul est réparti sur plusieurs processeurs (peut-être sur un réseau). L'application directe de la définition mathématique de la multiplication matricielle donne un algorithme qui prend du temps de l'ordre de n 3 opérations de champ pour multiplier deux matrices n × n sur ce champ ((n 3) en notation grand O). De meilleures limites asymptotiques sur le temps nécessaire pour multiplier les matrices sont connues depuis l' algorithme de Strassen dans les années 1960, mais on ne sait toujours pas quel est le temps optimal (c'est-à-dire quelle est la complexité de calcul de la multiplication matricielle). En décembre 2020, l'algorithme de multiplication matricielle avec la meilleure complexité asymptotique s'exécute dans un temps O(n 2.3728596) , donné par Josh Alman et Virginia Vassilevska Williams , mais cet algorithme est un algorithme galactique en raison des grandes constantes et ne peut pas être réalisé pratiquement. Algorithme itératif La définition de la multiplication matricielle est que si C = AB pour une matrice n × m A et une matrice m × p B , alors C est une matrice n × p avec des entrées c je j = ?? k = 1 m une je k b k j . {\displaystyle c_{ij}=\sum _{k=1}^{m}a_{ik}b_{kj}.} À partir de là, un algorithme simple peut être construit qui boucle sur les indices i de 1 à n et j de 1 à p , en calculant ce qui précède à l'aide d'une boucle imbriquée : Entrée : matrices A et B Soit C une nouvelle matrice de taille appropriée Pour i de 1 à n : Pour j de 1 à p : Soit somme = 0 Pour k de 1 à m : Ensemble somme ← somme + A ik × B kj Ensemble C ij ← somme Retour C Cet algorithme prend le temps Θ(nmp) (en notation asymptotique). Une simplification courante aux fins de l' analyse des algorithmes consiste à supposer que les entrées sont toutes des matrices carrées de taille n × n , auquel cas le temps d'exécution est Θ(n 3) , c'est-à-dire cubique dans la taille de la dimension. Comportement du cache Illustration de l'ordre principal des lignes et des colonnes Les trois boucles de la multiplication matricielle itérative peuvent être échangées arbitrairement les unes avec les autres sans effet sur l'exactitude ou le temps d'exécution asymptotique. Cependant, l'ordre peut avoir un impact considérable sur les performances pratiques en raison des modèles d'accès à la mémoire et de l' utilisation du cache de l'algorithme ; quel ordre est le meilleur dépend également du fait que les matrices sont stockées dans l' ordre principal des lignes , l' ordre principal des colonnes ou un mélange des deux. En particulier, dans le cas idéalisé d'un cache entièrement associatif composé de M octets et b octets par ligne de cache (c'est-à-direM/blignes de cache), l'algorithme ci-dessus est sous-optimal pour A et B stockés dans l'ordre des lignes principales. Quand n >M/b, chaque itération de la boucle interne (un balayage simultané d'une ligne de A et d'une colonne de B) entraîne un échec de cache lors de l'accès à un élément de B . Cela signifie que l'algorithme encourt Θ(n 3) d' absence de cache dans le pire des cas. À partir de 2010, la vitesse des mémoires par rapport à celle des processeurs est telle que les manques de cache, plutôt que les calculs réels, dominent le temps d'exécution des matrices de taille importante. La variante optimale de l'algorithme itératif pour A et B dans la disposition en ligne principale est une version en mosaïque , où la matrice est implicitement divisée en carreaux carrés de taille √ M par √ M : Entrée : matrices A et B Soit C une nouvelle matrice de taille appropriée Choisissez une taille de tuile T = Θ(√ M) Pour I de 1 à n par pas de T : Pour J de 1 à p par pas de T : Pour K de 1 à m par pas de T : Multiplier A I : I + T , K : K + T et B K : K + T , J : J + T en C I : I + T , J : J + T , soit : Pour i de I à min(I + T , n) : Pour j de J à min(J + T , p) : Soit somme = 0 Pour k de K à min(K + T , m) : Ensemble somme ← somme + A ik × B kj Ensemble C ij ← C ij + somme Retour C Dans le modèle de cache idéalisé, cet algorithme n'encourt que Θ(n 3/b √ M) les échecs de cache ; le diviseur b √ M revient à plusieurs ordres de grandeur sur les machines modernes, de sorte que les calculs réels dominent le temps de fonctionnement, plutôt que les défauts de cache.



Fait par Papa alassane SOW © étudiant en Administrateur et Maintenance en Réseaux Télé-informatique (AMRT) de bambey © 2014

Les outils de base en l'algorithmique

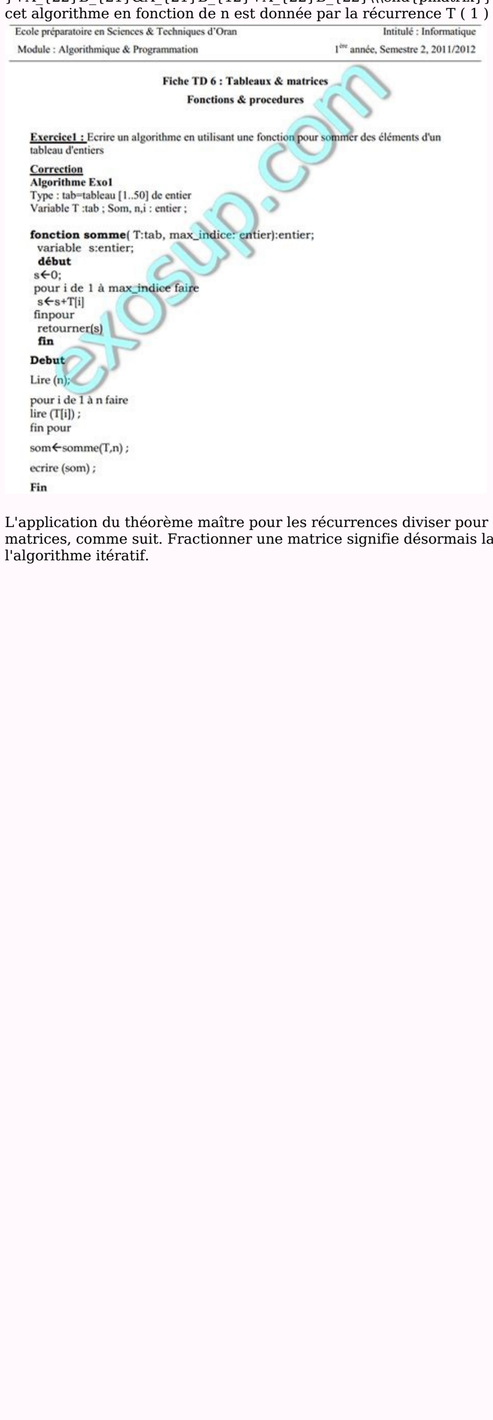
Exercice 1 :
Ecrire un **algorithme** qui demande les coordonnées de deux points dans le plan,
calcule et affiche à l'écran la distance entre ces deux points.
N.B
la distance entre deux points A(x1,y1) et B(x2,y2) est : AB= sqrt((x2-x1)^2 + (y2-y1)^2)
on donne la fonction sqrt(x) qui renvoie la racine carrée d'un nombre réel x .

Solution :
Algorithme calcul_distance;
Var
X1,x2,y1,y2,s : réels ;
Debut
Ecrire('entrer la valeur de x1 : ') ;
Lire(x1) ;
Ecrire(' entrer la valeur de y1 :') ;
Lire(y1) ;
Ecrire(' entrer la valeur de x2 : ') ;
Lire(x2) ;
Ecrire(' entrer la valeur de y2 : ') ;
Lire(y2) ;
S :=sqrt((x2-x1)^2 + (y2-y1)^2) ;
Ecrire('la distance entre A(',x1,',',y1,') et B(',x2,',',y2,') est : ',s) ;
Fin

Exercice 2 :
Elaborer un **algorithme** permettant de demander les valeurs de trois résistances r1,r2 et
r3 et de calculer et afficher leurs résistances équivalente dans les deux cas suivants :
N.B
Lorsque ces résistances sont branchées en série :
Rser = r1+r2+r3
Lorsque ces résistances sont branchées en parallèle :
Rpar=(r1*r2*r3)/(r1*r2+r1*r3+r2*r3)

Solution :

Algorithme de division pour régner Une alternative à l'algorithme itératif est l'algorithme diviser pour régner pour la multiplication matricielle. Cela repose sur le partitionnement par blocs C = (C 11 C 12 C 21 C 22) , UNE = (UNE 11 UNE 12 UNE 21 UNE 22) , B = (B 11 B 12 B 21 B 22) , {\displaystyle C= {\begin{pmatrix}C_{11}&C_{12}\\C_{21}&C_{22}\\\end{pmatrix}},\,A={\begin{pmatrix}A_{ 11}&A_{12}\\A_{21}&A_{22}\\\end{pmatrix}},\,B={\begin{pmatrix}B_{11}&B_{12}\\B_{21}&B_ {22}\\\end{pmatrix}},} qui fonctionne pour toutes les matrices carrées dont les dimensions sont des puissances de deux, c'est-à-dire que les formes sont 2 n × 2 n pour certains n . Le produit matriciel est maintenant (C 11 C 12 C 21 C 22) = (UNE 11 UNE 12 UNE 21 UNE 22) (B 11 B 12 B 21 B 22) = (UNE 11 B 11 + UNE 12 B 21 UNE 11 B 12 + UNE 12 B 22 UNE 21 B 11 + UNE 22 B 21 UNE 21 B 12 + UNE 22 B 22) {\displaystyle {\begin{pmatrix}C_{11}&C_{12}\\C_{21}&C_{22}\\\end{pmatrix}}={\begin{pmatrix}A_{11}&A_{12} \\A_{21}&A_{22}\\\end{pmatrix}}{\begin{pmatrix}B_{11}&B_{12}\\B_{21}&B_{22}\\\end{pmatrix}} ={\begin{pmatrix}A_{11}B_{11}+A_{12}B_{21}&A_{11}B_{12}+A_{12}B_{22}\\A_{21}B_{11 }+A_{22}B_{21}&A_{21}B_{12}+A_{22}B_{22}\\\end{pmatrix}}} qui consiste en huit multiplications de paires de sous-matrices, suivies d'une étape d'addition. L'algorithme diviser pour régner calcule les plus petites multiplications de manière récursive , en utilisant la multiplication scalaire c 11 = a 11 b 11 comme cas de base. La complexité de cet algorithme en fonction de n est donnée par la récurrence T (1) = ?? (1) ; {\displaystyle T(1)=\Thêta (1);} T (m) = 8 T (m / 2) + ?? (m 2) , {\displaystyle T(n)=8T(n/2)+\Thêta (n^{2}),} compte des huit appels récursifs sur des matrices de taille n /2 et (n 2) pour additionner les quatre paires de matrices résultantes élément par élément.



Ecole préparatoire en Sciences & Techniques d'Oran — Intitulé : Informatique
Module : Algorithmique & Programmation — 1ère année, Semestre 2, 2011/2012

Fiche TD 6 : Tableaux & matrices
Fonctions & procedures

Exercice1 : Ecrire un algorithme en utilisant une fonction pour sommer des éléments d'un tableau d'entiers

Correction

```
Algorithme Exo1
Type : tab=tableau [1..50] de entier
Variable T :tab ; Som, n,i : entier ;

fonction somme( T:tab, max_indice: entier):entier;
  variable  s:entier;
  début
 s←0;
 pour i de 1 à max_indice faire
  s←s+T[i]
 finpour
  retourner(s)
  fin
Debut
Lire (n);
pour i de 1 à n faire
lire (T[i]) ;
fin pour
som←somme(T,n) ;
ecrire (som) ;
Fin
```


L'application du théorème maître pour les récurrences diviser pour régner montre que cette récursivité a la solution Θ(n 3) , la même que l'algorithme itératif. Matrices non carrées Une variante de cet algorithme qui fonctionne pour des matrices de formes arbitraires et est plus rapide en pratique divise les matrices en deux au lieu de quatre sous-matrices, comme suit. Fractionner une matrice signifie désormais la diviser en deux parties de taille égale, ou de tailles aussi proches que possible de l'égalité dans le cas de dimensions impaires. Entrées : matrices A de taille n × m , B de taille m × p . Cas de base : si max(n , m , p) est inférieur à un certain seuil, utilisez une version déroulée de l'algorithme itératif.

Notions d'algorithme

Table des matières

1 Introduction **2**
1.1 Algorithme . . . 2
1.2 Conventions pour écrire un algorithme . . . 2
1.3 Types d'instructions . . . 3

2 Les Instructions **4**
2.1 Création d'un programme . . . 4
2.2 Lecture et affichage d'une variable . . . 4
2.3 Variable . . . 4
2.4 Affectation d'une variable . . . 5

3 Les tests **6**

1

Cas récursifs : Si max(n , m , p) = n , divisez A horizontalement : C = (UNE 1 UNE 2) B = (UNE 1 B UNE 2 B) {\displaystyle C={\begin{pmatrix}A_{1}\\A_{2}\end{pmatrix}}{B}={\begin{pmatrix}A_{1}B\\A_{2}B\ fin{pmatrix}}} Sinon, si max(n , m , p) = p , divisez B verticalement : C = UNE (B 1 B 2) = (UNE B 1 UNE B 2) {\displaystyle C=A{\begin{pmatrix}B_{1}&B_{2}\end{pmatrix}}={\begin{pmatrix}AB_{1}&AB_{2}\end{pmatrix}}} Sinon, max(n , m , p) = m . Divisez A verticalement et B horizontalement : C = (UNE 1 UNE 2) (B 1 B 2) = UNE 1 B 1 + UNE 2 B 2 {\displaystyle C={\begin{pmatrix}A_{1}&A_{2}\end{pmatrix}}
{\begin{pmatrix}B_{1}\\B_{2}\end{pmatrix}}=A_{ 1}B_{1}+A_{2}B_{2}} Comportement du cache Le taux d'échec du cache de la multiplication matricielle récursive est le même que celui d'une version itérative en mosaïque , mais contrairement à cet algorithme, l'algorithme récursif ignore le cache : aucun paramètre de réglage n'est requis pour obtenir des performances de cache optimales, et il se comporte bien dans un environnement de multiprogrammation où les tailles de cache sont effectivement dynamiques en raison d'autres processus occupant de l'espace de cache. (L'algorithme itératif simple est également inconscient du cache, mais beaucoup plus lent en pratique si la disposition de la matrice n'est pas adaptée à l'algorithme.) Le nombre d'échecs de cache subis par cet algorithme, sur une machine avec M lignes de cache idéal, chacune de taille b octets, est borné par ?? (m + m + p + m m + m p + m p b + m m p b M) {\displaystyle \Theta \left(m+n+p+{\frac {mn+np+mp}{b}}+{\frac {mnp}{b{\sqrt {M}}}}\right)} Algorithmes sous-cubiques Amélioration des estimations de l'exposant ω au cours du temps pour la complexité de calcul de la multiplication matricielle . O (m ??) {\displaystyle O(n^{\omega })} Il existe des algorithmes qui offrent de meilleurs temps d'exécution que les simples. Le premier à être découvert était l'algorithme de Strassen , conçu par Volker Strassen en 1969 et souvent appelé « multiplication matricielle rapide ». Elle repose sur une manière de multiplier deux matrices 2 × 2 qui ne nécessite que 7 multiplications (au lieu des 8 habituelles), au prix de plusieurs opérations d'addition et de soustraction supplémentaires. L'application récursive donne un algorithme avec un coût multiplicatif de . L'algorithme de Strassen est plus complexe, et la stabilité numérique est réduite par rapport à l'algorithme naïf, mais il est plus rapide dans les cas où n > 100 environ et apparaît dans plusieurs bibliothèques, comme BLAS . Il est très utile pour les grandes matrices sur des domaines exacts tels que les corps finis , où la stabilité numérique n'est pas un problème.
O (m Journal 2 ?? 7) ?? O (m 2.807) {\displaystyle O(n^{\log _{2}7})\approx O(n^{2.807})} C'est une question ouverte en informatique théorique dans quelle mesure l'algorithme de Strassen peut être amélioré en termes de complexité asymptotique . L' exposant de multiplication de matrice , généralement noté , est le plus petit nombre réel pour lequel toute matrice sur un champ peut être multipliée ensemble à l'aide d' opérations de champ. La meilleure liaison actuelle est celle de Josh Alman et Virginia Vassilevska Williams . Cet algorithme, comme tous les autres algorithmes récents dans cette ligne de recherche, est une généralisation de l'algorithme de Coppersmith-Winograd, qui a été donné par Don Coppersmith et Shmuel Winograd en 1990. L'idée conceptuelle de ces algorithmes est similaire à l'algorithme de Strassen : un moyen est conçu pour multiplier deux k × k -matrices avec moins de k 3 multiplications, et cette technique est appliquée de manière récursive. Cependant, le coefficient constant caché par la notation Big O est si grand que ces algorithmes ne valent la peine que pour les matrices trop grandes pour être traitées sur les ordinateurs actuels. ?? {\displaystyle \omega } m × m {\style d'affichage n\ fois n} m ?? + o (1) {\displaystyle n^{\omega +o(1)}} ??
{\displaystyle \omega } ?? < 2.3728596 {\displaystyle \omega <2.3728596} L'algorithme de Freivalds est un algorithme de Monte Carlo simple qui, étant donné les matrices A , B et C , vérifie en temps Θ(n 2) si AB = C .
Algorithmes parallèles et distribués L' algorithme diviser pour régner esquissé précédemment peut être parallélisé de deux manières pour les multiprocesseurs à mémoire partagée . Celles-ci sont basées sur le fait que les huit multiplications matricielles récursives dans (UNE 11 B 11 + UNE 12 B 21 UNE 11 B 12 + UNE 12 B 22 UNE 21 B 11 + UNE 22 B 21 UNE 21 B 12 + UNE 22 B 22) {\displaystyle {\begin{pmatrix}A_{11}B_{11}+A_{12}B_{21}&A_{11}B_{12}+A_{12}B_{22}\\A_{21}B_{11}+A_{22}B_{21}&A_{21}B_{12}+A_{22}B_{22}\\\end{pmatrix}}} peuvent être effectuées indépendamment les unes des autres, de même que les quatre sommations (bien que l'algorithme doive « joindre » les multiplications avant de faire les sommations). En exploitant tout le parallélisme du problème, on obtient un algorithme qui peut être exprimé en pseudocode de style fork-join : Procédure multiplier(C , A , B) : Cas de base: si n = 1 , ensemble c 11 ← un 11 × b 11 (ou multiplier une petite matrice de bloc). Sinon, allouez de l'espace pour une nouvelle matrice T de forme n × n , puis : Divisez A en A 11 , A 12 , A 21 , A 22 . Divisez B en B 11 , B 12 , B 21 , B 22 . Divisez C en C 11 , C 12 , C 21 , C 22 . Divisez T en T 11 , T 12 , T 21 , T 22 .
Exécution parallèle : Fourchette multiplie (C 11 , A 11 , B 11) . Multipliez la fourchette (C 12 , A 11 , B 12) . Fourchette multiplie (C 21 , A 21 , B 11) . Multipliez la fourchette (C 22 , A 21 , B 12) . Multipliez la fourchette (T 11 , A 12 , B 21) . Multipliez la fourchette (T 12 , A 12 , B 22) . Multipliez la fourchette (T 21 , A 22 , B 21) . Multipliez la fourchette (T 22 , A 22 , B 22) . Join (attendre la fin des fourches parallèles). ajouter(C , T) .
Désallouer T . La procédure add(C , T) ajoute T dans C , élément par élément : Cas de base: si n = 1 , ensemble c 11 ← c 11 + t 11 (ou faire une boucle courte, peut - être déroulé). Autrement: Divisez C en C 11 , C 12 , C 21 , C 22 . Divisez T en T 11 , T 12 , T 21 , T 22 . En parallèle: Fourche ajouter (C 11 , T 11) .
Fourche ajouter (C 12 , T 12) . Fourche ajouter (C 21 , T 21) . Fourchette ajouter (C 22 , T 22) . Rejoindre . Ici, fork est un mot-clé qui signale qu'un calcul peut être exécuté en parallèle avec le reste de l'appel de fonction, tandis que join attend que tous les calculs précédemment "fourchus" soient terminés.
partition atteint son objectif par la manipulation du pointeur uniquement. Cet algorithme a une longueur de chemin critique de (log 2 n) pas, ce qui signifie qu'il prend autant de temps sur une machine idéale avec un nombre infini de processeurs ; par conséquent, il a une accélération maximale possible de (n 3 /log 2 n) sur n'importe quel ordinateur réel. L'algorithme n'est pas pratique en raison du coût de communication inhérent au déplacement des données vers et depuis la matrice temporaire T , mais une variante plus pratique permet d' accélérer Θ(n 2) , sans utiliser de matrice temporaire. Multiplication matricielle par blocs.
Dans l'algorithme 2D, chaque processeur est responsable d'une sous-matrice de C . Dans l'algorithme 3D, chaque paire de sous-matrices de A et B qui est multipliée est affectée à un processeur.
Algorithmes évitant la communication et distribués Sur les architectures modernes à mémoire hiérarchique, le coût de chargement et de stockage des éléments de matrice d'entrée a tendance à dominer le coût de l'arithmétique. Sur une seule machine, il s'agit de la quantité de données transférées entre la RAM et le cache, tandis que sur une machine multi-nœuds à mémoire distribuée, il s'agit de la quantité transférée entre les nœuds ; dans les deux cas, il s'agit de la bande passante de communication . L'algorithme naïf utilisant trois boucles imbriquées utilise une bande passante de communication (n 3) . L'algorithme de canon , également connu sous l' algorithme 2D , est un algorithme d'évitement communication que les cloisons de chaque matrice d'entrée dans une matrice de blocs dont les éléments sont des sous - matrices de taille √ M / 3 par √ M / 3 où M est la taille de la mémoire rapide. L'algorithme naïf est ensuite utilisé sur les matrices de blocs, calculant les produits des sous-matrices entièrement en mémoire rapide. Ceci réduit la largeur de bande de communication vers O (n 3 / √ M) , qui est asymptotiquement optimal (pour les algorithmes performants Ω (n 3) de calcul). Dans un environnement distribué avec p processeurs agencés en un √ p par √ p maillage 2D, une sous - matrice du résultat peut être attribué à chaque processeur, et le produit peut être calculée avec chaque transmission de processeur O (n 2 / √ p) mots, ce qui est asymptotiquement optimal en supposant que chaque nœud stocke le minimum d'éléments O (n 2 / p) . Cela peut être amélioré par l' algorithme 3D, qui organise les processeurs dans un maillage cubique 3D, attribuant chaque produit de deux sous-matrices d'entrée à un seul processeur. Les sous-matrices de résultats sont ensuite générées en effectuant une réduction sur chaque ligne. Cet algorithme transmet O (n 2 / p 2/3) mots par processeur, ce qui est asymptotiquement optimal. Cependant, cela nécessite de répliquer chaque élément de matrice d'entrée p 1/3 fois, et nécessite donc un facteur de p 1/3 de mémoire de plus que ce qui est nécessaire pour stocker les entrées. Cet algorithme peut être combiné avec Strassen pour réduire davantage le temps d'exécution. Les algorithmes "2.5D" offrent un compromis continu entre l'utilisation de la mémoire et la bande passante de communication. Sur les environnements informatiques distribués modernes tels que MapReduce , des algorithmes de multiplication spécialisés ont été développés. Algorithmes pour les maillages Multiplication matricielle réalisée en 2n-1 étapes pour deux matrices n×n sur un maillage câblé. Il existe une variété d'algorithmes de multiplication sur des maillages .
Pour la multiplication de deux n × n sur un maillage bidimensionnel standard en utilisant l' algorithme de 2D Cannon , on peut compléter la multiplication en 3 n -2 étapes bien que cela soit réduit à la moitié de ce nombre pour les calculs répétés. Le tableau standard est inefficace car les données des deux matrices n'arrivent pas simultanément et doivent être complétées par des zéros. Le résultat est encore plus rapide sur un maillage à fils croisés à deux couches, où seulement 2 n -1 étapes sont nécessaires. Les performances s'améliorent encore pour les calculs répétés conduisant à une efficacité de 100 %. Le réseau à mailles câblées peut être considéré comme un cas particulier d'une structure de traitement non plane (c'est-à-dire multicouche). Voir également Les références Lectures complémentaires Ecrire l'algorithme effectuant le décalage des éléments d'un tableau Ecrire l'algorithme qui calcule le produit de deux matrices carées réelles A = CorrectionTD2.pdf Exercice 1 : Ecrire un algorithme qui permet de lire les valeurs d'un tableau de 50 entiers puis de calculer la somme de ses éléments Utiliser algo-chapitre-4.pdf Chapitre 8 : tableaux à deux dimensions Correction des exercices de travaux dirigés Exercice 1 : notes (somme de lignes et de colonnes) corr_tds.pdf Exercices Corrigés d'Algorithmique – 1ére Année MI 5 EXERCICE 1 Ecrire un algorithme qui demande un nombre à l'utilisateur, puis calcule et affiche le mi1an_algo-exercices_corriges.pdf Ateliers : Exercices corrigés En utilisant les tableaux, écrire un algorithme qui permet la saisie d'une Exercice 06 : Produit de deux matrices Atelier%2003_Operations%20sur%20les%20tableaux.pdf Exercices Corrigés Matrices Exercice 1 – Considérons les matrices `a 1) En utilisant l'algorithme du cours, montrer que la matrice suivante est EC3.17-18%20.pdf Exercice 1: Mise à zéro de la diagonale principale d'une matrice 2) Ecrire une analyse et un algorithme d'une fonction intitulée Somme_colonnne qui SERIE-RECURRENTS-CORIGE.pdf Exercice 1 On considère deux manières de représenter ce que l'on appelle des « matrices creuses », c'est-à-dire des matrices d'entiers contenant environ td4.pdf Exercice sur les tableaux à une dimension : écrire un programme permettant chaque case de la matrice en affichant « X » si la valeur de la case est Correction-Sujet_TDTP1_2.pdf 2) Ecrire la matrice transposée At de A et donner son format Exercice n° 3 1) Donner une matrice dont la transposée est égale à son opposée 2) Donnez la matrice A telle que pour tout indice i et j avec, 1 3? ?i et 1 3? ?j , le terme aij soit donné par la formule a i jij = ?2 Exercice n° 4 On donne 2 5 3 1 A = matricesexoscorriges.pdf Exercice 11 {) Appliquer avec pr ecision l'algorithme du cours pour inverser la matrice : M= 2 1 3 2 2M 2;2(R) : Pr eciser une expression de M 1, puis de Mcomme produit de matrices el ementaires Exercice 12 { Soit Aet Bdeux matrices carr ees de m^eme ordre, on suppose que la matrice EC3.15-16.pdf Matrice: Variable portant un nom et divisée en cases alignées sur deux axes Une matrice peut être considérée comme le regroupement de plusieurs tableaux de même taille Regrouper trois tableaux ayant chacun 7 éléments aboutira à une matrice de 3 lignes (tableaux) et 7 colonnes (Cases) algo-chapitre-4.pdf Les Structures de Contrôle (Conditionnelles – Itératives) Exercices Corrigés d'Algorithmique – 1ére Année MI 5 EXERCICE 1 Ecrire un algorithme qui demande un nombre à l'utilisateur, puis calcule et affiche le carré de ce nombre mi1an_algo-exercices_corriges.pdf Exercice 2 : Donner une repr esen tation du m^eme graphe (ci-dessus) en matrice d'incidence Quels probl emes rencontre-t-on? Exercice 3 : Proposer un algorithme de construction de la matrice d'incidence a partir de la liste d'adjacence d'un graphe, puis a partir de sa matrice d'adjacence TD1.pdf 2 Exercice 2 Ecrire l'algorithme e?ectuant le d¶ecalage des ¶el¶ements d'un tableau Exemple : † Tableau initial D E C A L A G E † Tableau modi?¶e (d¶ecalage µa gauche) E C A L A G E D Proc¶edure Decalage_gauche (T: Tableau de caractµeres, N: entier) VAR tmp: caractµere i: entier Debut tmp CorrectionTD2.pdf Exercice 1 : Mise en bouche (7 points) (a)(1 point) Deux nombres sont oppos es si leur somme est egale a 0 Deux nombres sont inverses si leur produit est egal a 1 Ecrire un algorithme sontInvOuOpp(a,b) ou a et b sont deux nombres, qui retourne Vrai si a et b sont inverses ou oppos es, Faux sinon Solution: Deux solutions parmi d'autres corrige.pdf