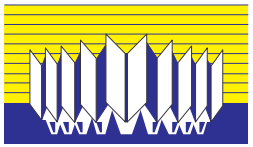
REPUBLIQUE TUNISIENNE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR,
DE LA TECHNOLOGIE ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE DE TUNIS ELMANAR



FACULTE DES SCIENCES DE TUNIS
DEPARTEMENT DES SCIENCES DE L'INFORMATIQUE

PROJET DE FIN D'ETUDES
D'INGENIEURS EN INFORMATIQUE

Etude d'Algorithmes Parallèles de Data Mining

organisme d'accueil : Faculté des Sciences de Tunis.

Réalisé par : Heithem ABBES.
Majed CHATTI.

Encadré par : Mr. Mohamed JEMNI.
Mme. Khadija ARROUR.
Mr. Yahia SLIMANI.

ANNEE UNIVERSITAIRE 2002/2003

Plan

- Introduction.
- Algorithmes séquentiels.
- Algorithmes parallèles.
- Implémentations parallèles.
- Evaluation expérimentale.
- Conclusion et perspectives.

Introduction

Introduction

- Augmentation des volumes des bases de données
- Extraction d'informations implicites.
- Emergence de nouvelles techniques.
  - Knowledge Discovery in Databases (KDD).
  - Data Mining, Text Mining, Web Mining, Multimedia Mining.
- Intérêts du Data Mining :
  - Marketing, Médecine …

➥ **Algorithmes séquentiels de recherche de règles associatives.**

- Problèmes de performances

➥ **Nouvelles approches parallèles.**

Algorithmes séquentiels

- ☒ Règles Associatives.
- ☒ Algorithme Apriori.
- ☒ Exemple.
- ☒ Conclusion.

Règles associatives

- Règles associatives (Agrawal).
  - A $\Longrightarrow$ B
  - Exemple:
    - **Age > 25 & Revenu < 8000 $\Longrightarrow$ modèle_voiture=populaire**
- Concepts de base:
  - Support : nombre d'apparitions / nombre de transactions.
    - Soit *supmin* le support minimum accepté.
  - Support de la règle : sup(A $\Longrightarrow$ B) = sup(A U B)
  - Confidence : pourcentage des transactions contenant B sachant qu'elles contiennent A.
  - Confidence de la règle : conf(A $\Longrightarrow$ B) = sup (A U B) / sup(A)
- Recherche de Règles associatives.
- Décomposition du problème:
  - Générer l'ensemble des items fréquents.
  - Déduire les règles associatives.

Algorithme Apriori

- L'algorithme *Apriori (Agrawal)*
  - Déterminer l'ensemble des *1-itemsets_fréquents*.
  - Générer l'ensemble des *k-itemsets_candidats* à partir de l'ensemble des *(k-1)-itemsets_fréquents*.
  - Calculer les supports des items candidats.
  - Déterminer l'ensemble des *k-itemsets_fréquents* (*minsup*).
- L'algorithme *AprioriTID*
  - Représente une amélioration de l'algorithme *Apriori*.
  - Moins d'accès à la base de données.
  - Remplacer la base de données par un ensemble intermédiaire.

Exemple (1/3)

$minsup = 2$

Tid	Items
1	A C D E
2	AB C
3	A B C
4	ABE
5	BE
6	BC

Base de données

Exemple (2/3)

- Calcul des supports des *1-itemset_candidats*.

Items	A	B	C	D	E
Support	4	5	4	1	3

- Génération de l'ensemble des *1-itemsets_frequents*.

Items				
Support				

- Génération de l'ensemble des 2-*itemsets_candidats*.

Items	AB	AC	AE	BC	BE	CE
Support	3	3	2	3	2	1

Exemple (3/3)

Items	AB	AC	AE	BC	BE	CE
Support	3	3	2	3	2	1

- Génération de l'ensemble des 2-*itemsets_frequents*.

Items					
Support					

- Génération de l'ensemble des 3-*itemsets_candidats*.
  - Génération de l'ensemble {ABC, ABE, ACE, BCE}.
  - Elimination des items : ACE, BCE car CE n'est pas fréquent.

Items	ABC	ABE
Support	2	1

- Génération de l'ensemble des 3-*itemsets_frequents* = **ABC**.
- Ensemble des candidats = Ø $\Longrightarrow$ FIN.

Conclusion

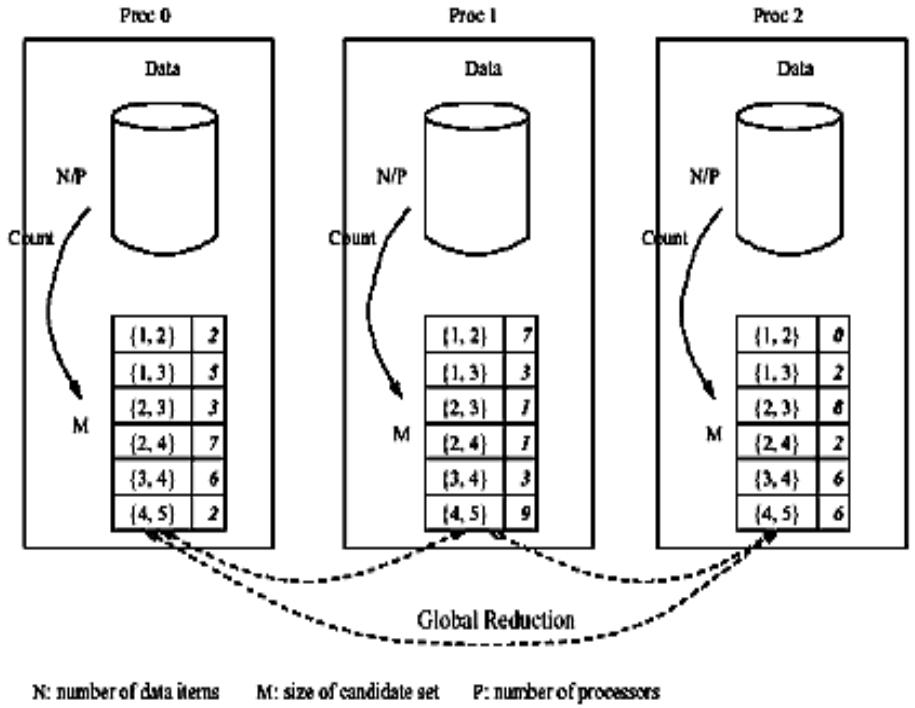
- Caractéristiques des algorithmes séquentiels
  - Coût Exponentiel
  - Plusieurs scans de la base de données
  - Très mauvaises performances
- Solution
  - Parallélisme ?

Algorithmes parallèles

- Count Distribution (CD).
- Data Distribution(DD).
- Intelligent Data Distribution(IDD).

Count Distribution (CD) (1/2)

- Agrawal & Shafer.
- Répartition de la base de données sur les processeurs.
- Pas de division de l'ensemble des candidats.
- Redondance de traitement.
  - Accès à la base.
  - Calcul des supports.
- Communication à la fin du calcul.
- Opération de réduction globale.



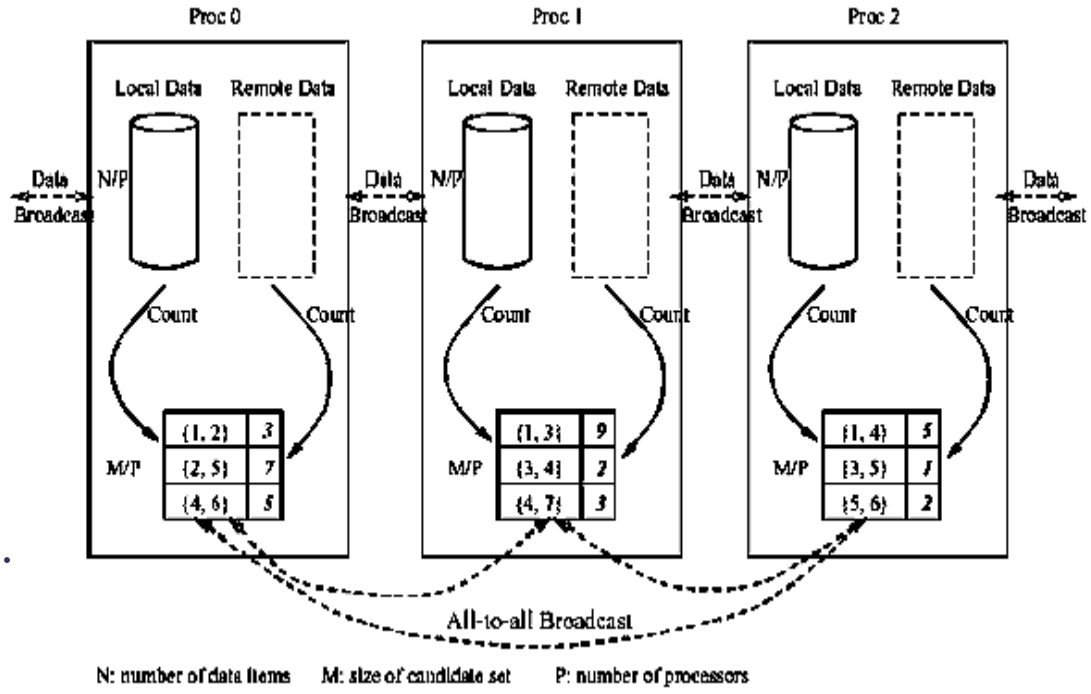
Count Distribution (CD)

Count Distribution (CD) (2/2)

- Etape préliminaire : générer l'ensemble des items fréquents.
- *Première étape* : générer l'ensemble des items candidats à partir de l'ensemble des items fréquents générés à l'itération précédente.
- *Deuxième étape* : calculer les support des items candidats.
- *Troisième étape* : échanger des supports calculés entre les différents processeurs.
- *Quatrième étape*: générer l'ensemble des items fréquents.
- *Cinquième étape* : terminer / revenir à la première étape.

Data Distribution (DD) (1/2)

- Agrawal & Shafer.
- Partitionne la base de données.
- Partitionne l'ensemble des candidats (Aléatoire).
- Echange de buffers de données.
- Communication de type ALL-TO-ALL.



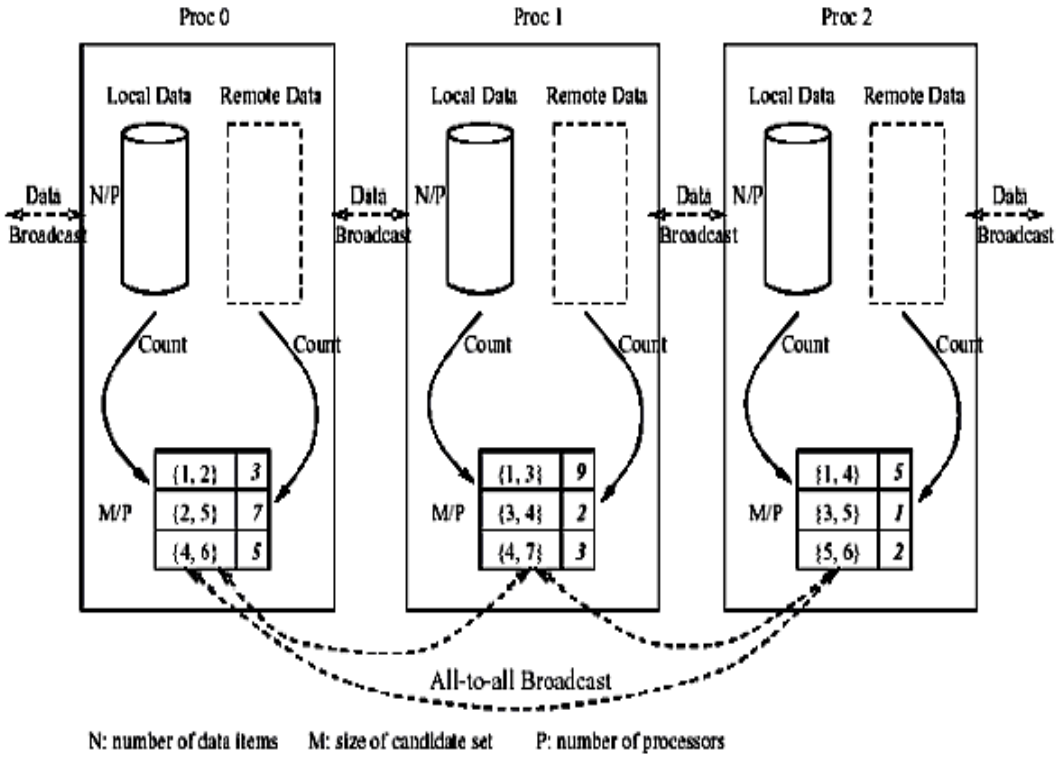
Data Distribution (DD)

Data Distribution (DD) (2/2)

- Etape préliminaire : générer de l'ensemble des items fréquents.
- *Première étape* : - générer l'ensemble items candidats.
  - répartir cet ensemble sur les processeurs.
- *Deuxième étape* : calculer les supports des candidats locaux.
- *Troisième étape* : former l'ensemble local des items fréquents.
- *Quatrième étape* : échanger ces ensembles entre les processeurs.

Intelligent Data Distribution (IDD)

- Amélioration de DD.
- Processus de communication en anneau logique.



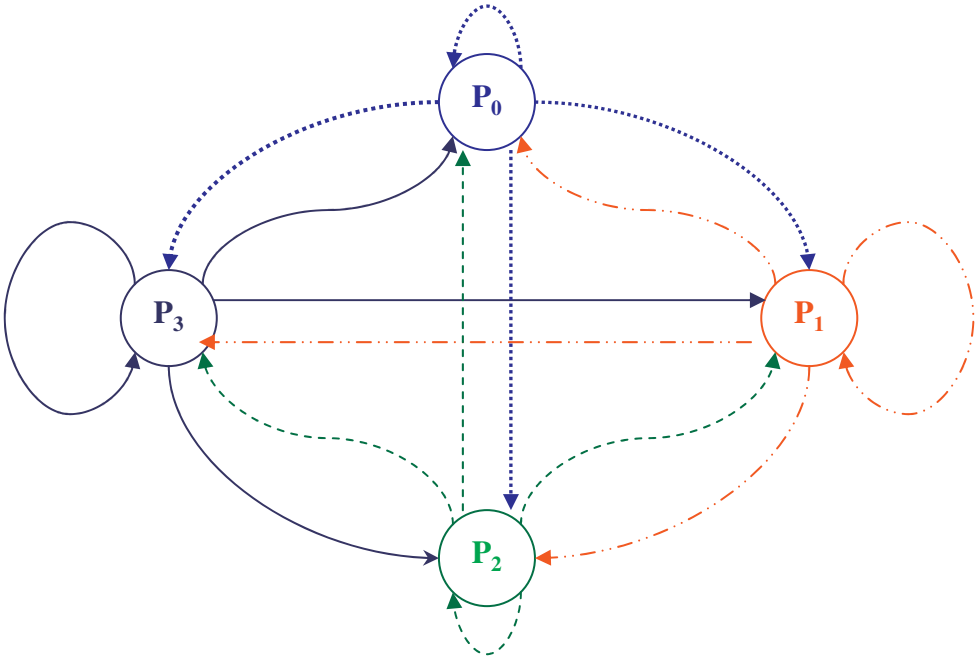
Intelligent Data Distribution (IDD)

Implémentations parallèles

- Data Distribution.
  - Processus de communication.
  - Programme.
- Intelligent Data Distribution
  - Processus de communication.
  - Programme.

Data Distribution (1/4)

- Processus de communication:
  - Chaque processeur transmet ses données à tous les autres.

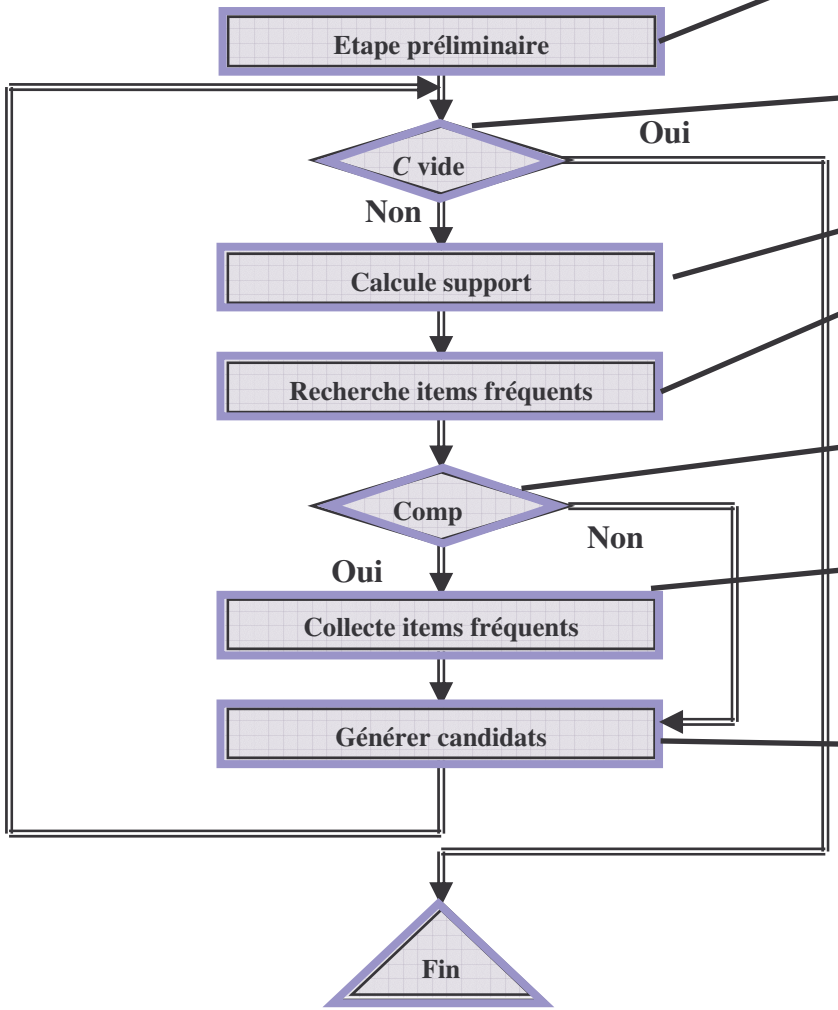


Communication ALL_TO_ALL

- Quatre processeurs P_0,P_1,P_2,P_3
- P_0 transmet ses données à P_1,P_2,P_3.
- P_1 transmet ses données à P_0,P_2,P_3.
- P_2 transmet ses données à P_0,P_1,P_3.
- P_3 transmet ses données à P_0,P_1,P_2.

Data Distribution (2/4)

- Programme:



Déterminer les items distincts, leurs supports, puis éliminer les items non fréquents et générer l'ensemble candidat e taille 2.

candidats.

Accède à la base locale et les autres bases pour calculer les supports des items candidats

Con

supérieur au support minimum.

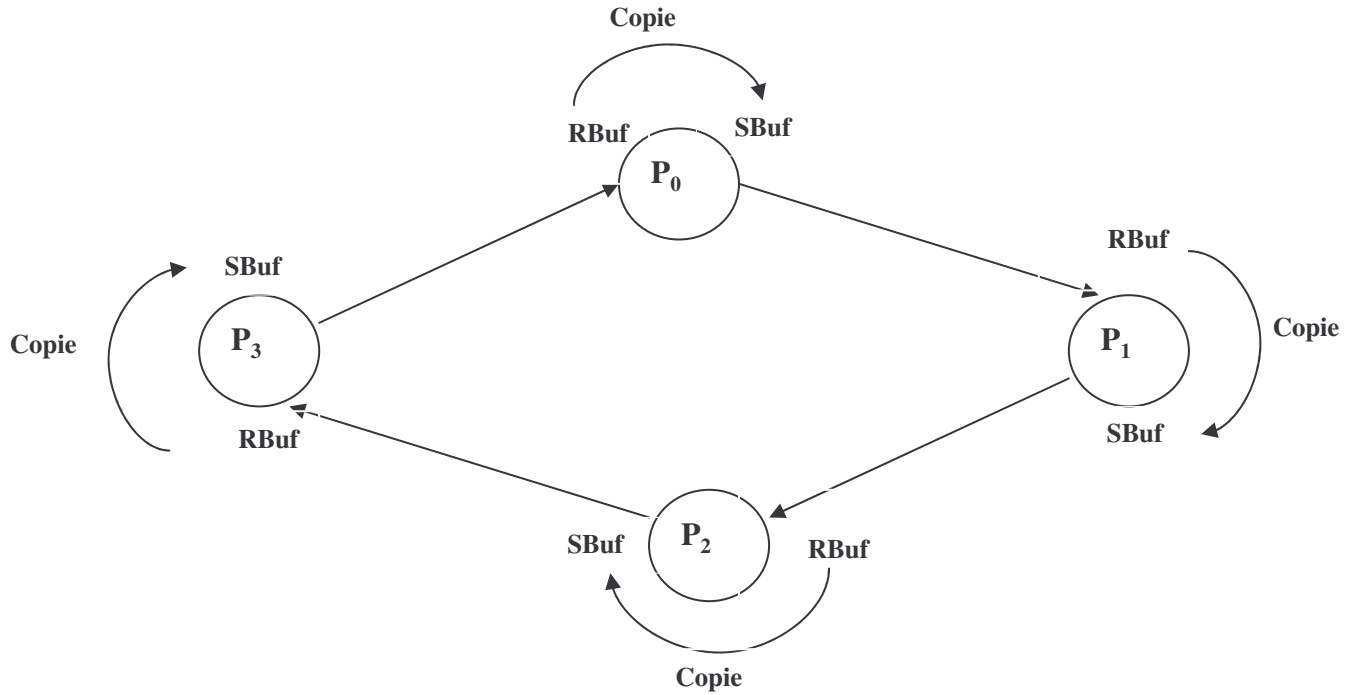
Ce test est vrai lorsque le nombre de candidats est supérieur au nombre de processeurs.

Chaque processeur transmet son calcul local à tous les autres pour construire l'ensemble global des items fréquents.

Génération de l'ensemble des candidats de taille k+1

Intelligent Data Distribution(1/2)

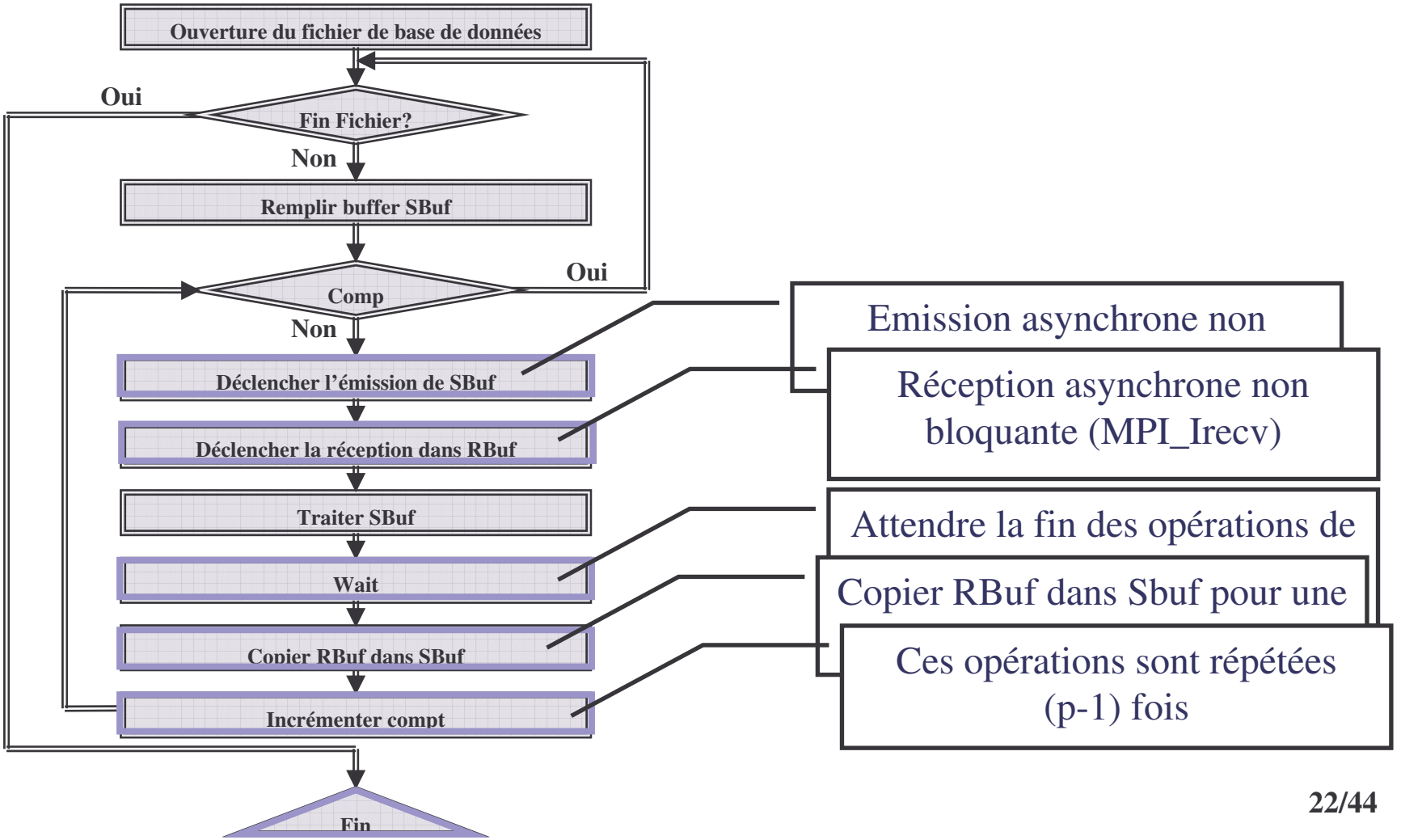
- Processus de communication:
  - Deux communications point à point : émission et réception.
  - Copie du contenu du buffer de réception dans le buffer d'émission.



Communication en anneau logique

Intelligent Data Distribution (2/2)

- Fonction calcul support:



Evaluation expérimentale

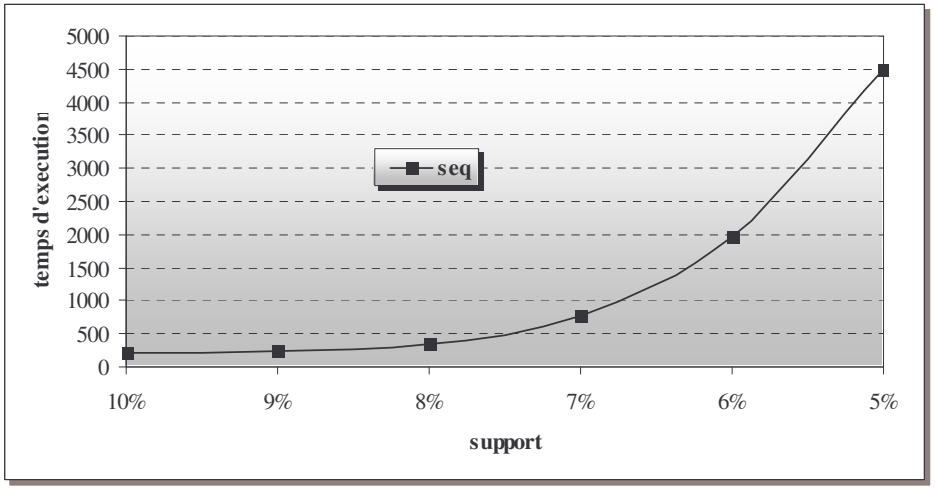
- Environnement de travail.
  - Machine parallèle SP2.
  - Bibliothèque MPI.
- Exécution séquentielle.
- Exécution parallèle de DD.
- Exécution parallèle de IDD.
- Comparaison entre DD et IDD.
- Synthèse.

Environnement de travail

- Machine parallèle SP2.
  - 8 nœuds.
  - 4 processeurs par nœuds.
  - Entre nœuds: mémoire distribuée.
  - Au sein d'un même nœud: mémoire partagée.
- Bibliothèque MPI
  - Interface de communication entre processus.
- Base de données synthétique d'IBM de 100000 transactions.

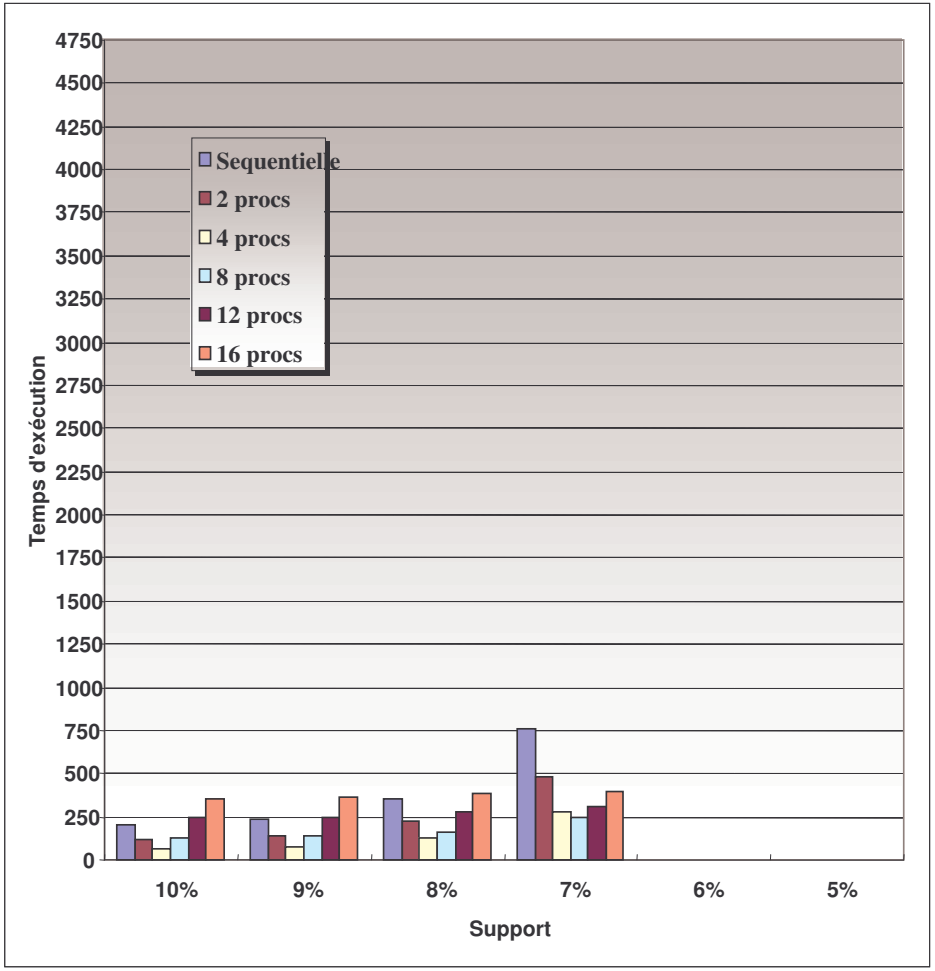
Exécution séquentielle

- DD/IDD sur un seul processeur.
- Nécessaire pour calculer les paramètres du parallélisme.
  - Accélération : temps séquentiel/temps parallèle. .
  - Efficacité : accélération/nombre de processeurs.



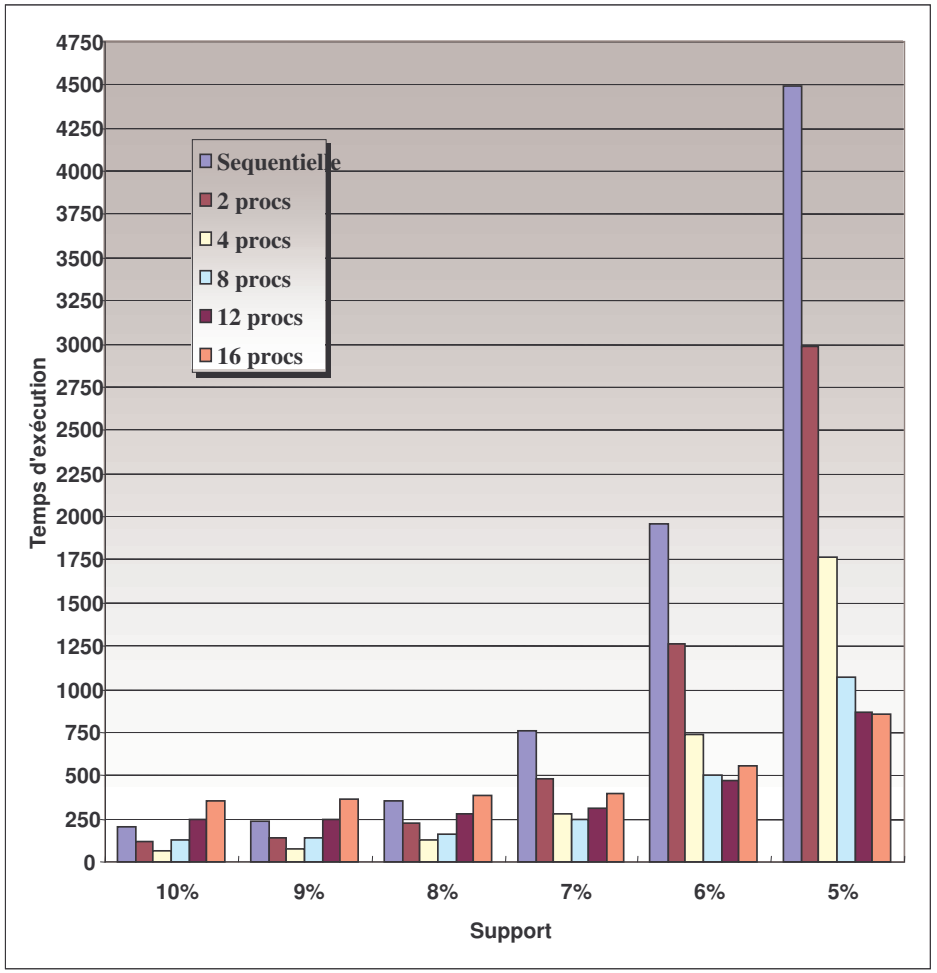
- Temps d'exécution augmente lorsque le support diminue.

Exécution parallèle de DD (1/2)



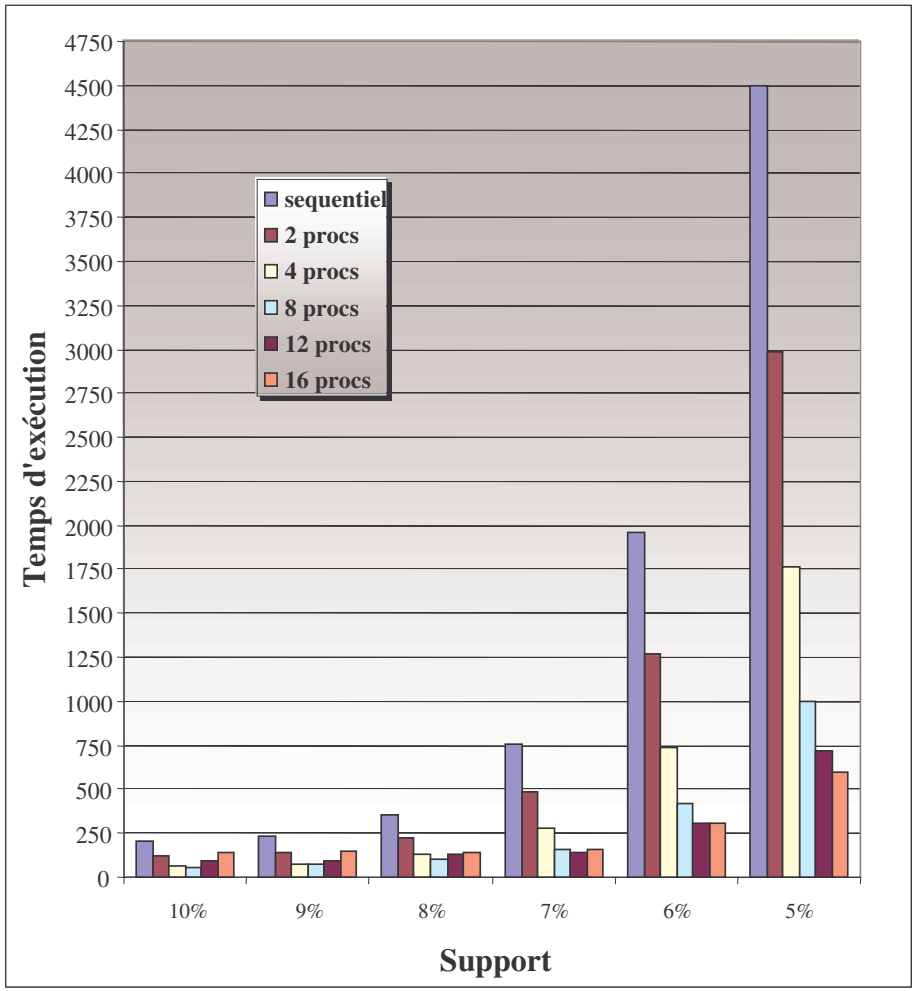
- **Supports 10%, 9%, 8% :**
  - communication importante/traitement.
  - pas de communication au sein d'un même nœud.
  - Meilleur temps d'exécution obtenu avec 4 processeurs.
- **Support 7% :**
  - Traitement augmente légèrement.
  - Apport du parallélisme non apparent
  - Meilleur résultat obtenu avec 8 processeurs.

Exécution parallèle de DD (2/2)



- **Support 6% :**
  - Part de traitement augmente.
  - Inutile de passer jusqu'à 16 processeurs.
  - Meilleur temps d'exécution obtenu avec 12 processeurs.
- **Support 5% :**
  - Apport du parallélisme est nettement perceptible.
  - Meilleur temps d'exécution obtenu avec 16 processeurs.

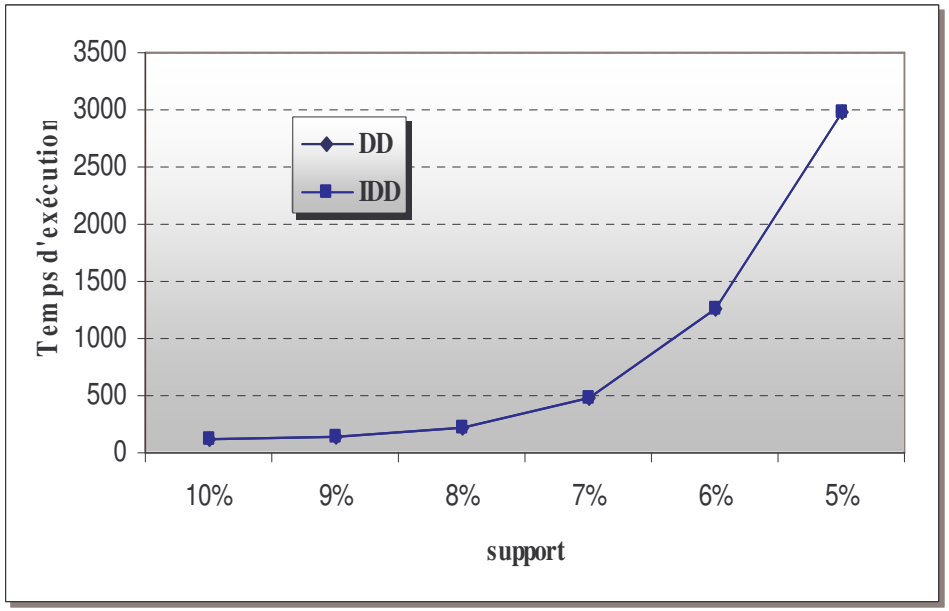
Exécution parallèle de IDD



- **Supports 10%, 9%, 8% :**
  - **Meilleur temps donné par 8 processeurs grâce au mode de communication.**
- **Support 7% :**
  - **Plus de traitement .**
  - **Meilleur temps de réponse donné par 12 processeurs**
- **Supports 6% et 5%:**
  - **l'exécution sur 16 processeurs donne le meilleur temps de réponse.**

Comparaison entre DD et IDD (1/14)

- Cas de deux processeurs:
  - **temps d'exécution:**



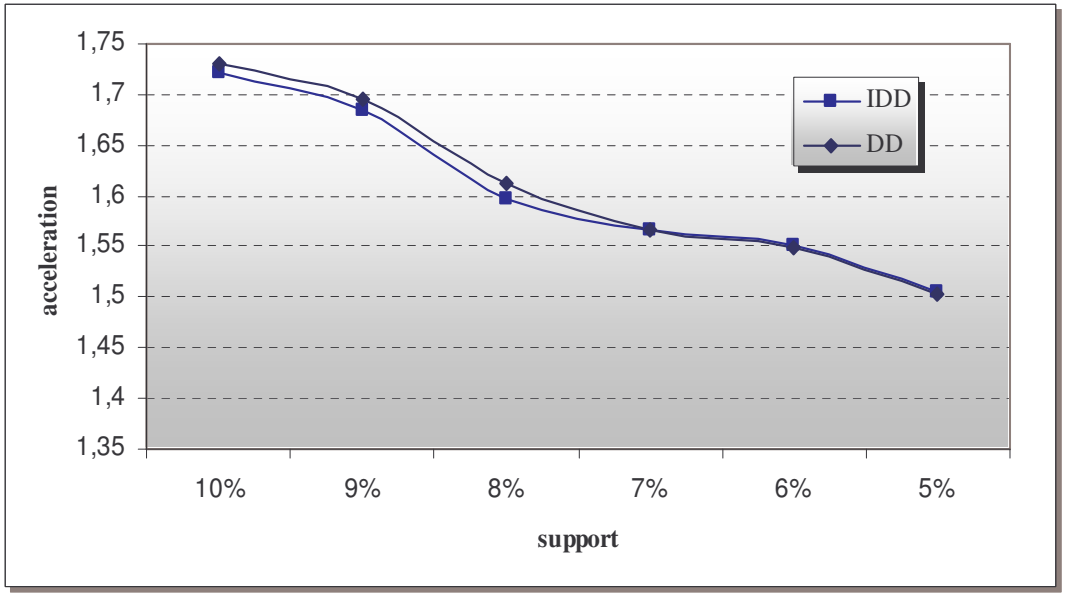
Courbes d'exécution de DD et IDD sur 2 processeurs

- **Deux processeurs de même nœud.**
- **Communication à travers la mémoire partagée**
- **Différence entre DD et IDD**
- **Processus de communication**

➥ **Même résultat pour DD et IDD.**

Comparaison entre DD et IDD (2/14)

- Cas de deux processeurs :
  - **Accélération :**



Courbes d'accélération de DD et IDD sur 2 processeurs

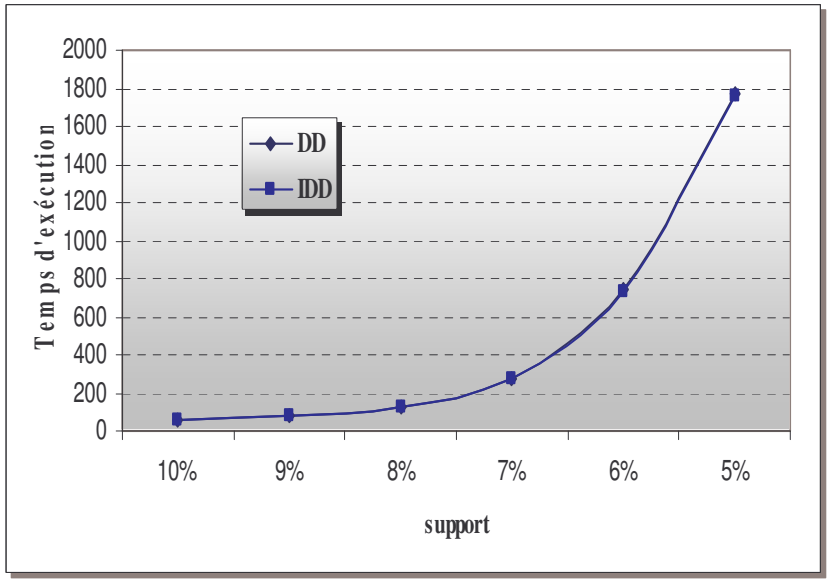
- **Support diminue => traitement augmente.**
- **Plus de synchronisations.**
- **accélération diminue légèrement.**
- **Même temps d'exécution pour DD et IDD.**

➥ **Même accélération pour DD et IDD**

Comparaison entre DD et IDD (3/14)

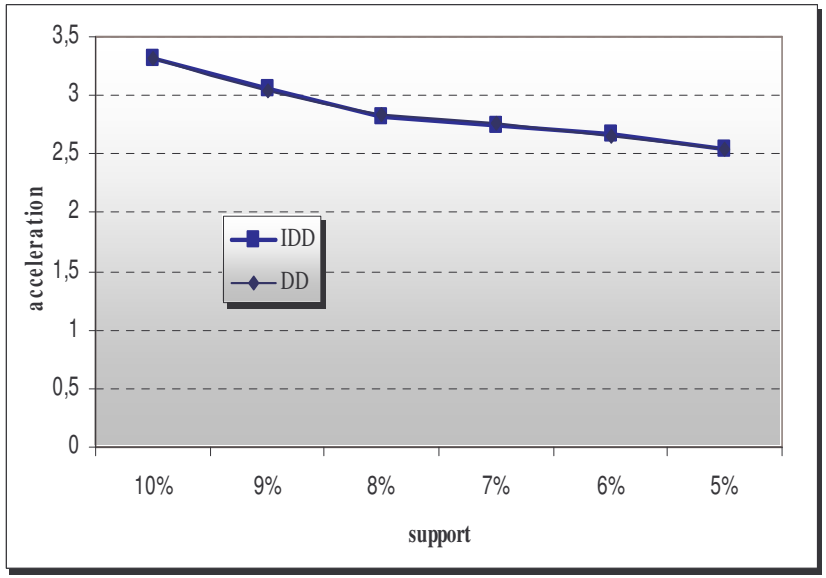
- **Cas de quatre processeurs:**

Temps exécution



Courbes d'exécution de DD et IDD sur 4 processeurs

Accélération

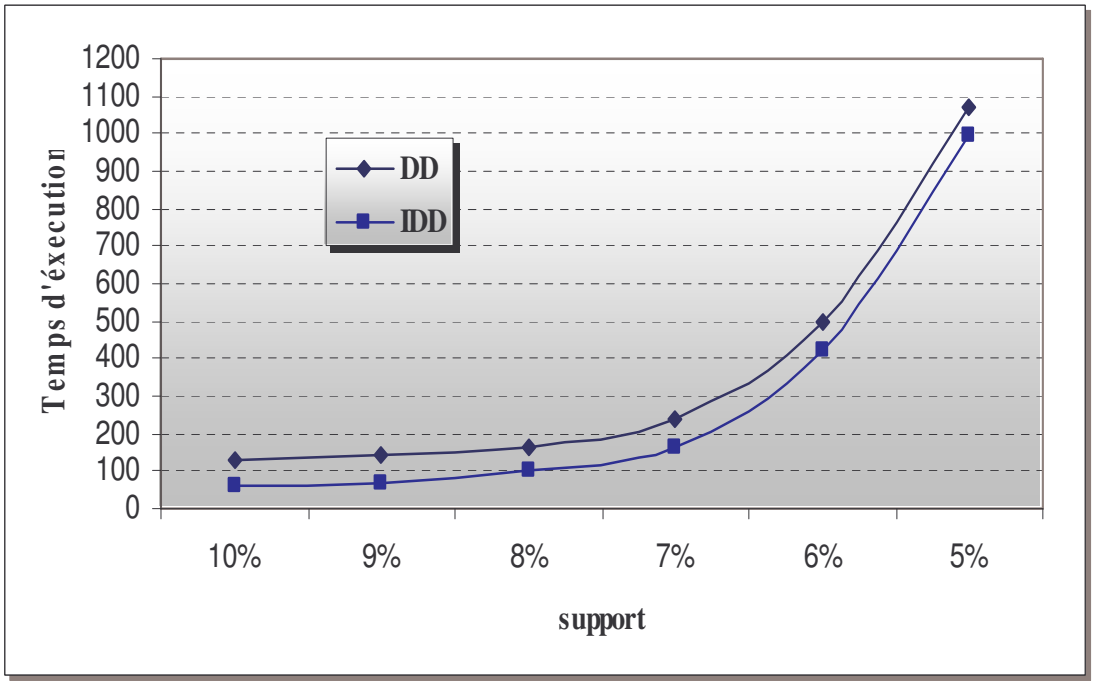


Courbes d'accélération de DD et IDD sur 4 processeurs

- Quatre processeurs de même nœud.
- Même constatations que sur 2 processeurs.
- Plus de synchronisations => décélération plus importante.

Comparaison entre DD et IDD (4/14)

- Cas de huit processeurs:
  - Temps d'exécution:



Courbes d'exécution de DD et IDD sur 8 processeurs

- IDD meilleur que DD
- Différence négligeable.

Comparaison entre DD et IDD (5/14)

- **Cas de huit processeurs:**
  - **Accélération:**

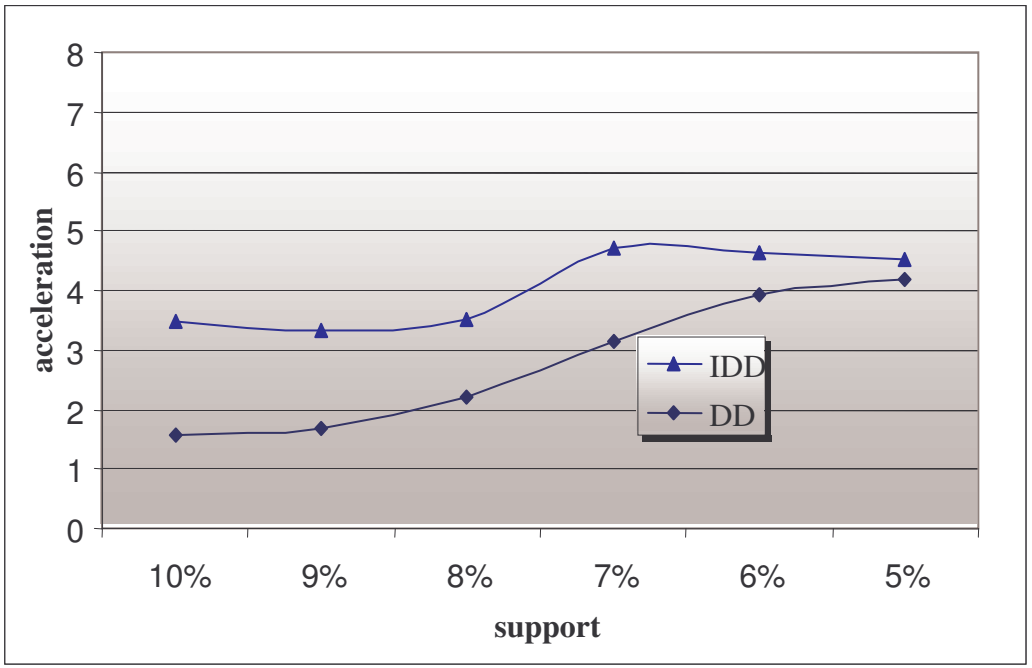
support	10%	9%	8%	7%	6%	5%
Accélération DD	1,59	1,67	2,19	3,14	3,93	4,19
Accélération IDD	3,47	3,32	3,53	4,73	4,62	4,51

Tableau d'accélérations du DD et IDD

- Pour les support élevés: DD donne des accélérations faibles.
- DD s'améliore pour les supports faibles.
- IDD donne de meilleures accélérations même sur 10% et 9%.
- Apport de IDD au niveau de la communication par rapport à DD.

Comparaison entre DD et IDD (6/14)

- Cas de huit processeurs:



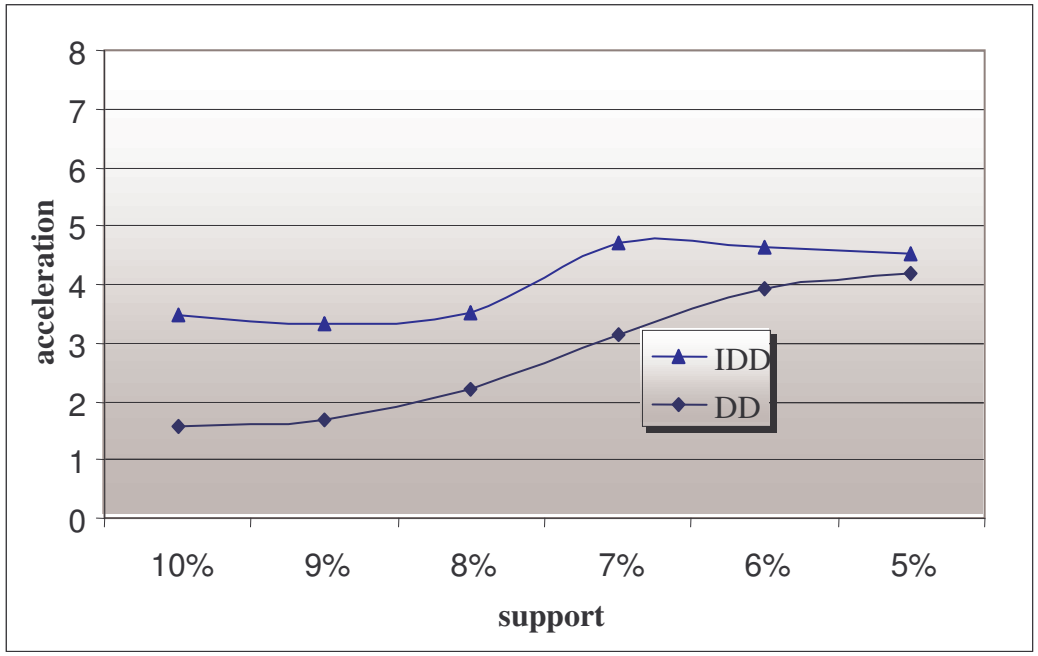
Courbes d'accélération de DD et IDD sur 8 processeurs

Pour IDD :

- Supports 10% , 9% , 8%:
  - Communication élevée.
  - Faible accélération.
- Support 7% :
  - Nombre de candidats augmente.
  - ➥ Traitement augmente.
  - ➥ Meilleure accélération.
- Supports 6% et 5% :
  - Légère diminution de l'accélération.
  - Nombre de processeurs insuffisants.

Comparaison entre DD et IDD (7/14)

- Cas de huit processeurs:



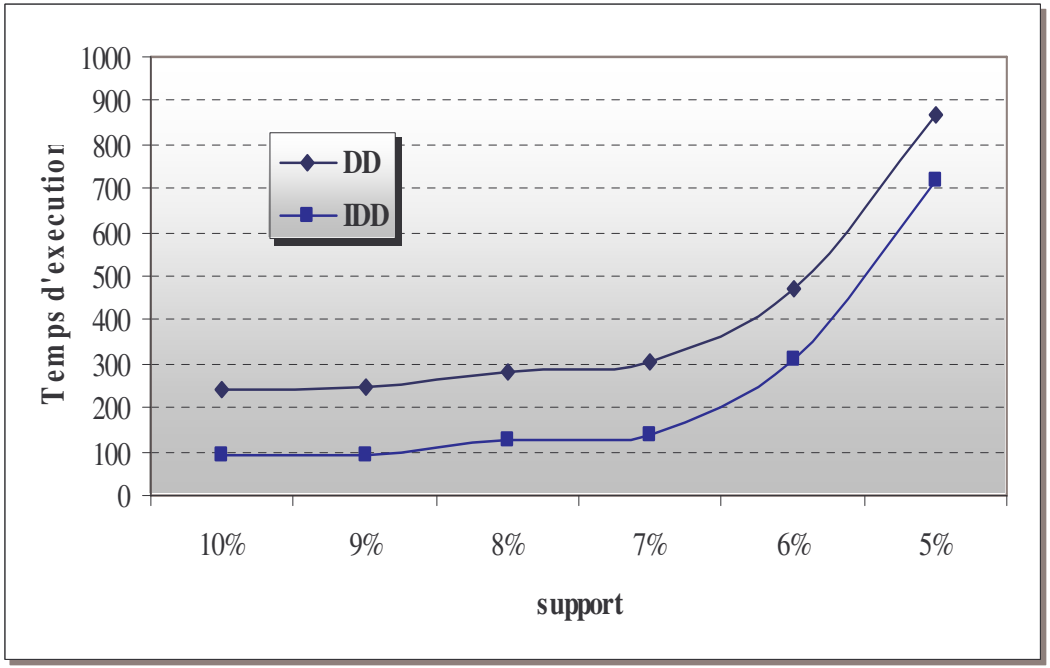
Courbes d'accélération de DD et IDD sur 8 processeurs

Pour DD :

- Processus de Communication non performant.
- Accélération médiocre pour les supports élevés.
- Support diminue =>Accélération augmente.
- Accélération ne dépasse jamais celle de IDD.

Comparaison entre DD et IDD (8/14)

- **Cas de douze processeurs:**
  - **Temps d'exécution:**



Courbes d'exécution de DD et IDD sur 12 processeurs

- **Plus de communication.**
- **La différence entre DD et IDD est plus importante.**

Comparaison entre DD et IDD (9/14)

- **Cas de douze processeurs:**
  - **Accélération:**

support	10%	9%	8%	7%	6%	5%
Accélération DD	0,85	0,95	1,27	2,47	4,15	5,19
Accélération IDD	2,26	2,52	2,79	5,43	6,32	6,27

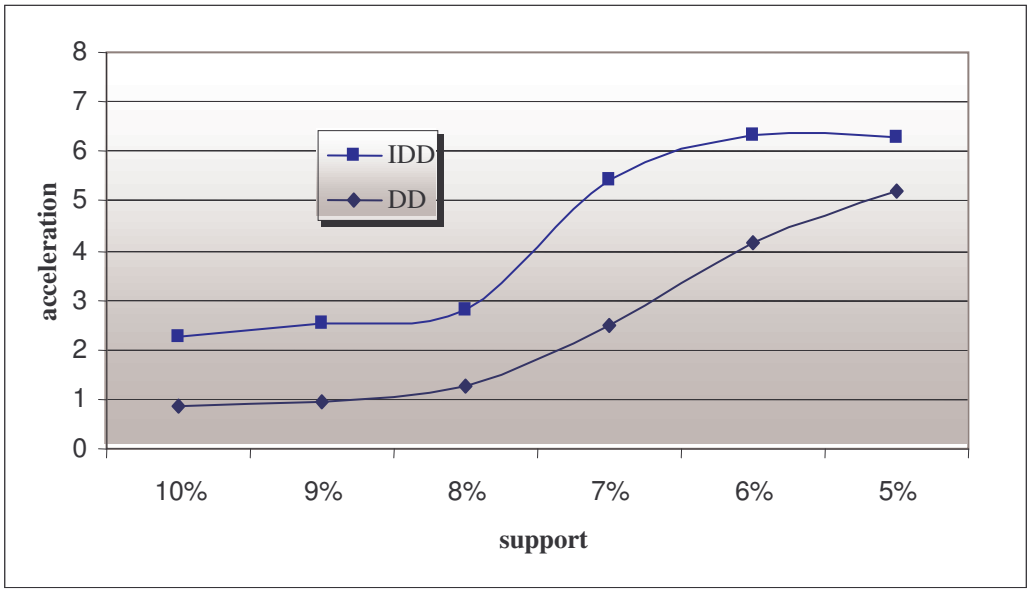
Tableau d'accélérations du DD et IDD

- DD donne des résultats médiocres
- DD moins performant que l'algorithme séquentiel pour 10% et 9%.
- IDD donne une faible accélération pour 10% et 9% mais nettement meilleure que celle de DD.
- Pour les supports 6% et 5% la différence entre DD et IDD diminue.

➔ La part de traitement est plus importante par rapport à celle de communication.

Comparaison entre DD et IDD (10/14)

- **Cas de douze processeurs:**
  - **Accélération:**



Courbes d'accélération de DD et IDD sur 12 processeurs

- Pour IDD :
  - Passage de 8% à 7%
  - Amélioration importante de l'accélération.
  - Nombre de processeurs insuffisants à partir de 6%.

  ➥ Légère atténuation de la courbe.

- Pour DD :
  - Support diminue=>Accélération augmente.
  - Accélération ne dépasse jamais celle de IDD

Comparaison entre DD et IDD (11/14)

- **Cas de seize processeurs :**
  - **Temps d'exécution :**

Support	10%	9%	8%	7%	6%	5%
Temps séquentiel	206,75	233,13	355,69	758,11	1960,99	4495,86
Temps parallèle DD	356,44	360,11	379,79	400,75	560,44	855,11
Temps parallèle IDD	138,74	145,49	144,56	159,47	310,54	600,89

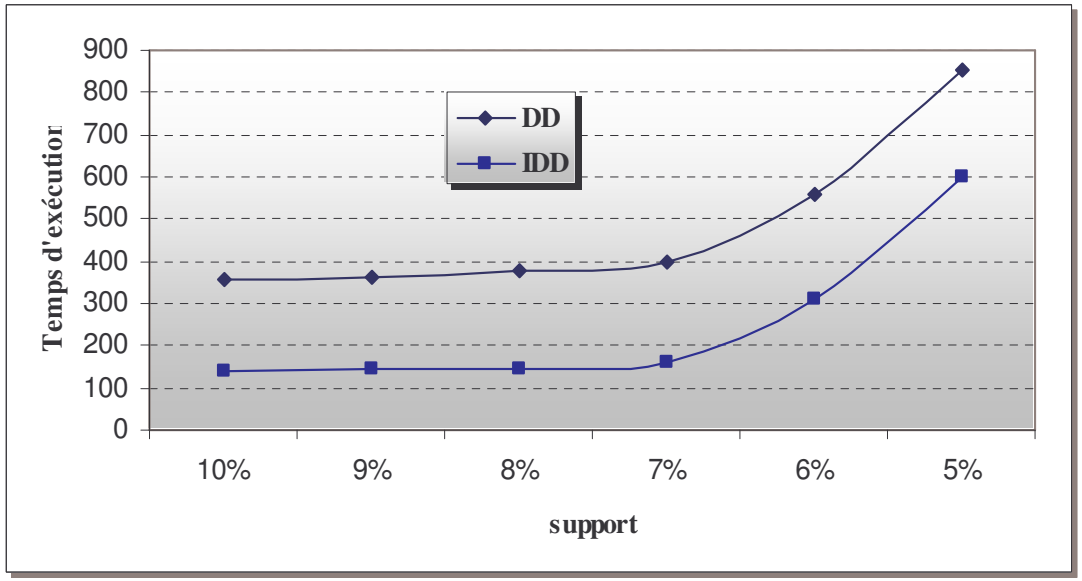
Tableau d'accélérations du DD et IDD

- DD est moins performant que l'algorithme séquentiel pour les supports suivants : 10%, 9% et 8%.

- IDD donne de mauvais temps de réponses pour ces supports mais ne dépasse pas le temps séquentiel.

Comparaison entre DD et IDD (12/14)

- Cas de seize processeurs :
  - Temps d'exécution :



Courbes d'exécution de DD et IDD sur 16 processeurs

- La différence entre DD et IDD est plus perceptible.

Comparaison entre DD et IDD (13/14)

- **Cas de seize processeurs :**
  - **Accélération :**

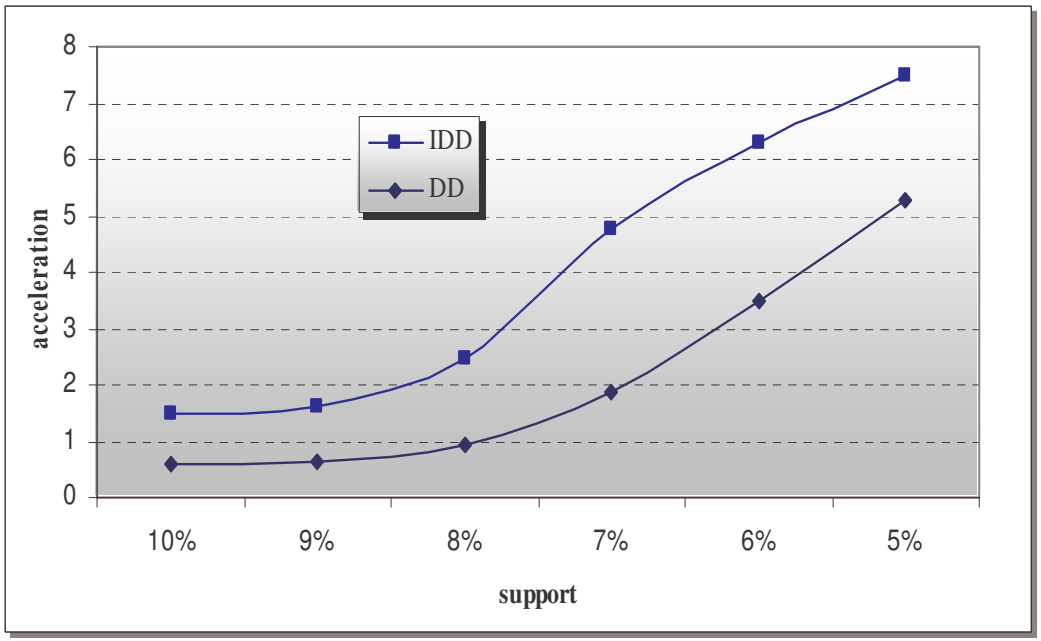
support	10%	9%	8%	7%	6%	5%
Accélération DD	0,58	0,65	0,94	1,89	3,50	5,26
Accélération IDD	1,49	1,60	2,46	4,75	6,31	7,48

Tableau d'accélérations du DD et IDD

- Sur 16 processeurs : apport net de IDD par rapport à DD.
- Pour 10% : accélération très faible.
- Pour 5% : bonne accélération.

Comparaison entre DD et IDD (14/14)

- **Cas de seize processeurs :**
  - **Accélération :**



Courbes d'accélération de DD et IDD sur 16 processeurs

- Pour IDD :
  - Pas d'atténuation de la courbe.
- Pour DD :
  - accélération s'améliore de plus en plus que le support diminue.
  - Ne dépasse jamais celle de IDD.

Synthèse

- Pour DD, avec des support élevés, les meilleurs temps de réponse sont donnés par l'exécution sur 4 processeurs.
- Pour des supports faibles, 4 processeurs ne suffisent plus.
- Si nous augmentons le nombre de processeurs sans diminuer le support, DD devient moins performant que l'algorithme séquentiel à partir de 12 processeurs.
- L'apport de IDD est d'autant plus perceptible que le support est faible et que le nombre de processeur est élevé.
- Les résultats données par IDD sont toujours plus meilleurs que ceux donnés par DD.

  ↪ **IDD est bien meilleur que DD.**

- Accès disque + réseaux non rapide

  ↪ **Dégradation des performances des deux algorithmes**

Conclusion et Perspectives

Conclusion :

- Les techniques de Data Mining suscitent de plus en plus d'intérêt.
- Les algorithmes séquentiel on un coût exponentiel.
- Les améliorations faites sur ces algorithmes ne sont pas intéressantes.
- Le recours à des techniques de parallélisme semble être la meilleure solution.
- Les algorithmes de recherche de règles associatives réalisent plusieurs passes sur la base de données.

Perspectives :

- Prévoir une nouvelle approche qui réduit les accès au disque.
- Prévoir une nouvelle Parallélisation de l'algorithme Apriori.

Etude d'Algorithmes Parallèles de Data Mining

Réalisé par
Heithem ABBES.
Majed CHATTI.

Encadré par
Mr. Mohamed JEMNI.
Mme. Khadija ARROUR.
Mr. Yahia SLIMANI

MERCI POUR VOTRE ATTENTION

Pour plus d'informations veuillez contacter:
a_heithem@yahoo.fr & c_majed@yahoo.fr

PROJET DE FIN D'ETUDES D'INGENIEURS EN INFORMATIQUE

Etude d'Algorithmes Parallèles de Data Mining

Réalisé par
Heithem ABBES.
Majed CHATTI.

Encadré par
Mr. Mohamed JEMNI.
Mme. Khadija ARROUR.
Mr. Yahia SLIMANI

MERCI POUR VOTRE ATTENTION

Pour plus d'informations veuillez contacter:
a_heithem@yahoo.fr & c_majed@yahoo.fr