Une synthèse bibliographique sur les disques RAID

Singhoff Frank

singhoff@cnam.fr

26 janvier 1996

Table des matières

Table des figures

Chapitre 1

Introduction

Durant les dix dernières années, on a constaté un progrès technologique important sur certaines parties des ordinateurs. En effet, l'industrie des semi-conducteurs a énormement progressé d'où une forte amélioration des performances des processeurs et des composants constituant les mémoires principales.

Toutefois, les systèmes informatiques ne peuvent être réduits aux deux seules composantes "processeur et mémoire principale" : en effet, les mémoires secondaires influent aussi dans les performances d'un ordinateur. Or, bien que mémoire et processeur aient bénéficié de progrès technologiques conséquents, il n'en est pas de même pour les mémoires secondaires, et en particulier des disques (en tout cas sur certains critères de performances).

Bien sûr, ces péripheriques ont eux aussi subi l'influence du progrès technologique : on note une forte augmentation de la densité des disques durs ainsi qu'une diminution de leur diamètre. En 1983, les disques avaient une taille de cinq pouces 1/4, aujourd'hui, on peut trouver des disques durs de 1,3 pouces seulement. Cette augmentation de la densité permet de mettre nettement plus d'informations sur ces disques de 1,3 pouces que sur les disques de cinq pouces 1/4 de l'époque ! On peut aussi remarquer une forte augmentation de la capacité de stockage (on trouve sur le marché pour les particuliers des disques de un giga octet et plus).

Les temps d'accès et la vitesse de rotations des disques ont, en proportion, nettement moins progressé. (Les temps d'accès sont passés de 20 ms en 1983 à 10 ms aujourd'hui et le nombre de rotations par minute, quant à lui, est passé de 3600 en 1980 à 5400 ou 7200 aujourd'hui [1])

1. Ces chiffres sont issus de l'article[5].

Ceci explique que les progrès sur les disques soient bien inférieurs à ceux obtenus sur les processeurs et les mémoires principales, et qu'ils ne suffisent pas à obtenir les performances nécessaires pour bénéficier pleinement de la rapidité des processeurs et des mémoires principales.

La loi d'Amdahl illustre parfaitement le fait que les performances d'un système soient bornées par l'élément le moins performant du système (bien que la loi d'Amdahl soit souvent utilisée pour la détermination de l'accélération dans les programmes parallèles qui possèdent une partie de code séquentiel, cette loi s'applique tout à fait dans notre cas).

En d'autres termes, pour pouvoir profiter des bonnes performances que l'on obtient sur les processeurs et la mémoire principale, il faut améliorer les performances des mémoires secondaires à un même degré.

C'est à ce niveau qu'interviennent les disques RAID [2] . Un disque RAID est un groupe de disques de faible volume, qui apparait aux applications comme un disque logique unique. Les disques RAID ne sont pas les seules techniques qui permettent d'améliorer les performances des systèmes informatiques. En effet, d'autres méthodes comme l'utilisation de caches mémoires, ou l'augmentation de la taille de la mémoire principale sont utilisées (pour minimiser les accès fichiers dans le cadre de mémoire virtuelle par exemple). Toutefois, ces techniques ne permettent pas d'améliorer les performances des entrées/sorties dans les systèmes transactionnels (où l'on effectue de nombreux accès concernant des données de faible taille) et dans les machines massivement parallèles (où l'on accède, par le biais d'un faible nombre de requêtes, à une quantité très importante d'informations). Les disques RAID trouvent donc leurs domaines d'applications dans ces deux cas de figure.

Ces deux justifications ne sont pas les seules à expliquer l'intérêt que l'on porte aujourd'hui aux disques RAID. En effet, le développement de la micro-informatique a joué un rôle déterminant : l'énorme marché créé par les micro-ordinateurs a permis de fortement diminuer le coût des disques de faible volume, et donc de rendre concurrentiels les disques RAID par rapport aux disques de grand volume (leur prix au méga octets étant sensiblement identique). Cette économie d'échelle a aussi permis d'en améliorer leur qualité.

Ainsi, dans le chapitre deux, nous étudierons les différents niveaux de

2. RAID pour "Redundant Arrays of Inexpensive disks".

disques RAID, nous examinerons leurs propriétés et leurs domaines d'applications. Puis, dans le chapitre trois, nous discuterons des problèmes de performances auxquels sont confrontés les disques RAID. Le chapitre quatre abordera quant à lui les problèmes de tolérance aux pannes. Enfin, avant de conclure dans le chapitre six, nous regarderons dans le chapitre cinq quelques exemples de prototypes à base des technologies RAID.

Chapitre 2

Les différentes architectures des disques RAID

Les disques RAID ont été conçus par une équipe de l'université de Berkeley. En fait, un disque RAID est un ensemble de disques de faible volume qui sont rassemblés en un seul disque logique de grand volume. L'intêret consiste à permettre une parallélisation des entrées/sorties, et donc une augmentation des performances.

Il existe aujourd'hui sept types de disques RAID (appelés "niveaux de disque RAID"). Au début du projet RAID, dans les premières publications qui furent [21] et [20] , seuls cinq niveaux furent spécifiés. Puis, les niveaux zéro et six furent présentés dans [5] et [13]. **La majeure partie des informations données dans ce chapitre se trouve dans ces quatre références clefs.**

Dans ce chapitre, nous détaillerons donc l'architecture de chacun de ces sept niveaux, en donnant à chaque fois leurs caractéristiques en terme de performances, résistance aux pannes, capacité de stockage et domaine possible d'utilisation. Toutefois, il faut signaler qu'en fait, il existe d'innombrables architectures de disques RAID, les sept que nous allons présenter sont les architectures généralement admises dans la littérature. Malgré cela, nous verrons dans le chapitre traitant des performances qu'il en existe bien d'autres, ayant bien sûr des propriétés et des domaines d'applications différents . Toutes les architectures de disques RAID sont en fait une combinaison de ces deux facteurs :

- La manière dont les informations de redondance sont distribuées sur l'ensemble des disques du tableau,

– Le type de redondance utilisé (Parité, code de Hamming ou code Reed Solomon).

Enfin, il faut signaler qu'il est difficile de dire quelle est LA meilleure architecture. En effet, la comparaison est ardue car il entre en ligne de compte un nombre important de critères qui peuvent dépendre des besoins et des contraintes des utilisateurs :

– le coût de l'architecture,

– et/ou le domaine d'application du RAID,

– et/ou le degré de fiabilité,

– et/ou si les applications peuvent accepter que le disque soit indisponible quelques instants après une panne (pour la reconstruction des informations perdues),

– et/ou les performances sur écriture et/ou lecture,

– et/ou le prix au méga octet

– etc.

2.1 Le niveau zéro des disques RAID

Ce niveau peut ne pas être classé dans les architectures RAID [1] car il ne comporte pas d'information de redondance. Il n'y a donc aucune protection contre les pannes d'un disque du tableau et donc, **une panne suffit pour perdre de l'information**.

Toutefois, cette absence de redondance lui confère un excellent rapport volume de stockage/coût, ainsi qu'une bonne rapidité d'écriture sur le disque puisqu'il n'y a aucune information de redondance à mettre à jour. De ce fait, cette architecture est essentiellement utilisée avec des machines massivement parallèles où l'on préfère avantager les performances au détriment de la fiabilité.

1. Nous nous permettrons toutefois de l'y mettre car de nombreux auteurs le considèrent comme tel.

Il est à préciser que les accès en lecture sont quant à eux, moins efficaces par rapport à certaines architectures comme celle que nous allons voir : les disques miroirs.

Un tableau de niveau zéro de quatre disques de données est réprésenté sur la figure 2.1. Dans nos figures, les zones en blanc sont les disques de données utilisateurs, les zones en gris sont les disques de redondances. Ici bien sûr, il n'y a pas de disque en gris !

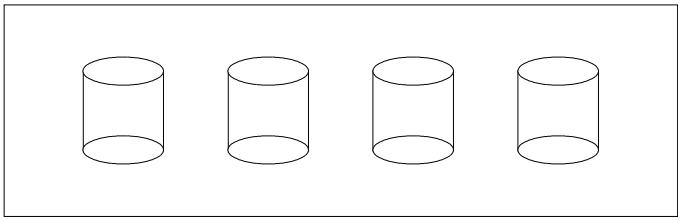

FIG. 2.1 – *Un tableau de niveau zéro*

2.2 Le niveau un des disques RAID

Ces disques sont aussi appelés "disques miroirs" car chaque disque de données utilisateurs est dupliqué par un disque de redondance (disque en gris sur la figure 2.2).

Les performances en écriture de ce type de tableau ne sont pas très bonnes : en effet chaque écriture sur un disque de données utilisateurs doit être répercutée **par une écriture identique sur le disque miroir** correspondant au disque de données, ce qui est très pénalisant, puisque dans ce cas, on écrit deux fois **l'ensemble** des données. Par contre, les performances en lecture sont bonnes car on peut utiliser, aussi bien les disques de données utilisateurs que les disques de redondance pour servir les requêtes. Ce qui permet de distribuer la charge sur l'ensemble des disques, qu'ils soient de redondance ou non.

Enfin, il faut ajouter qu'ils ont le coût le plus élevé de toutes les architectures que nous allons présenter, mais qu'en contre partie, ils possédent une

excellente résistance aux pannes.

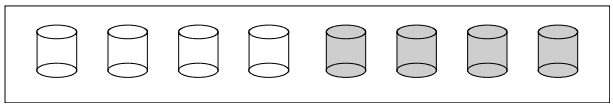

FIG. 2.2 – *Un tableau de disques de type "Miroir"*

Dans les niveaux un à cinq, il faut préciser que la panne d'un disque déclenche la reconstruction des données qu'il contenait et que pendant cette reconstruction, l'occurence d'une nouvelle panne engendre une perte de données.

2.3 Le niveau deux des disques RAID

Ce type de tableau de disques se base sur les codes de Hamming. Cette utilisation des codes de Hamming a été copiée sur celle qui en est faite dans les mémoires de type DRAM[2].

Petit rappel sur les codes Détecteurs et correcteurs d'erreurs. Soit deux mots binaires de n bits. On définit la distance de Hamming comme étant le nombre de bits qui différencie ces deux mots (en informatique, la distance de Hamming peut être calculée très simplement en effectuant un XOR (OU EXCLUSIF) entre les deux mots). Pour être capable de détecter "d" erreurs **simples** sur un mot, il faut que les mots du code aient une distance de Hamming de "d+1". De plus, pour être capable de corriger "d" erreurs simples, il faut alors une distance de "2d+1"[3].

Hamming a determiné une équation qui permet, en fonction de la taille du mot non codé, de déterminer le nombre de bits de redondances et la taille du mot du code. Soit :

$$n = m + r \tag{2.1}$$

où m est la taille du mot non codé en bits, r le nombre de bits de redoncances et n, la taille du mot du code.

2. DRAM pour Dynamic ramdom access memory.
3. Une description succinte des codes correcteurs peut être obtenue dans [26].

La formule de Hamming est alors :

$$m + r + 1 \leq 2^r \tag{2.2}$$

Sur notre exemple de la figure 2.3, si l'on ne fait qu'un contrôle de parité sur les disques RAID, c'est à dire, en ajoutant un seul disque de redondance, on voit tout de suite que l'on sera capable de détecter une erreur, mais non de la corriger ni de déterminer quel disque est faux !(en effet, les contrôles de parité ont une distance de Hamming de 2, grâce à "d+1", on voit que l'on peut détecter, mais la distance de 2 est inférieur à "2d+1", donc, on ne peut corriger). On est donc obligé d'utiliser un code de Hamming, et c'est l'inéquation 2.2 qui est utilisée pour déterminer le nombre de disque : dans notre exemple nous avons m = 4, or il faut que r = 3 pour que l'inéquation 2.2 soit vérifiée. Donc, il faut trois disques de redondance.

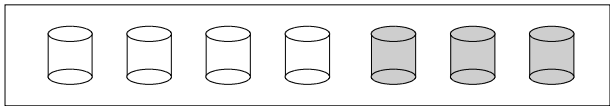

FIG. 2.3 – *Un tableau constitué par un code de Hamming*

Au niveau de la capacité de stockage, les RAID de niveau deux sont bien sûr plus interressants que les disques miroirs puisque le nombre de disques de redondances est inférieur. Toutefois, ce type de disque semble peu utilisé puisque les RAID de niveau trois offrent un même niveau de fiabilité pour des performances et un coût meilleurs. Ce type d'architecture est bien adaptée pour les machines fortement parallèles (d'après [21], la société Thinking Machine Corporation avait commercialisé un RAID de niveau deux batisé "Data Vault" et destiné à ses "Connection Machines".

2.4 Le niveau trois des disques RAID

Dans ce niveau, les données sont distribuées bit par bit sur l'ensemble des disques de données. Ainsi, une requête en écriture concernant vingt bits de données, écrira cinq bits par disque (si l'on reste toujours avec notre exemple de quatre disques de données). Cette répartition a pour effet qu'une seule requête d'entrées/sorties sur le tableau est effectuée en un intervalle de

temps : il n'y a pas d'accès parallèles sur des disques différents, concernant des requêtes différentes (il est à remarquer que cette contrainte existe aussi sur le niveau deux).

Comme on le voit sur la figure 2.4, la grande différence entre cette architecture et celle du niveau deux, c'est le nombre de disques de parité : celui-ci est passé à un, et ce quelque soit le nombre de disques de données. Dans ce niveau, on fait l'hypothèse que les contrôleurs peuvent déterminer quel est le disque en panne. Nous n'avons donc plus besoin d'un code de Hamming pour déterminer cette information. Un simple code de parité sur l'unique disque de parité suffit.

Il est à remarquer que si l'abaissement du nombre de disques de parité permet toujours de reconsituer les informations perdues au cours d'UNE panne, le fait qu'il y ait moins de disques au total, a pour effet d'améliorer la fiabilité générale du tableau. Les rapports entre nombre de disques et fiabilité du tableau seront abordés dans le chapitre traitant de la tolérance aux pannes. Enfin, les performances, sont équivalentes à celles des tableaux de niveau deux.

Puisque le niveau trois ne peut traiter qu'une seule requête d'entrées/sorties à la fois, il est inadapté au système transactionnel. Par contre, il convient parfaitement aux machines massivement parallèles nécessitant de gros débits d'entrées/sorties.

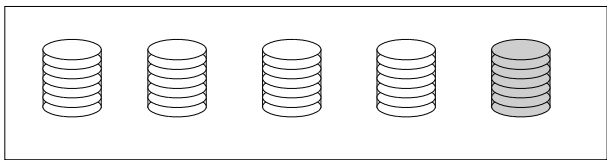

FIG. 2.4 – *Un tableau de disques de type entrelacement de bits*

2.5 Le niveau quatre des disques RAID

Le niveau quatre est proche du niveau trois mais avec une granularité plus forte en ce qui concerne la taille des données réparties. En effet, ici, on n'entrelace plus les données utilisateurs sur plusieurs disques de bit en bit,

mais de bloc en bloc (les blocs pouvant être des pistes, des secteurs ou des cylindres).

De plus, la méthode d'accès au tableau est différente : le niveau quatre permet des accès en parallèle de plusieurs requêtes. Une lecture dont la quantité d'informations à lire est plus petite que la taille des blocs entrelacés provoque une lecture d'un seul disque, et non plus une lecture sur tous les disques du tableau.

Si les accès en lecture ont un taux de parallélisation très fort, il en n'est pas de même pour les écritures : en effet, il faut toujours mettre à jour les blocs de parité. Toutefois, ici aussi on n'accéde plus à la totalité des disques : un accès en écriture nécessite la lecture de deux blocs et l'écriture de deux blocs. Avant l'écriture, on lit le bloc de données à mettre à jour ainsi que le bloc de parité, puis on écrit le bloc de données utilisateur, enfin on effectue le calcul de la formule 2.3 et on met à jour le bloc de parité (Cette opération est appelée "read-modify-write").

```
Nouveau bloc de parit\'e =   (Ancien bloc de donn\'ees
                        XOR  Nouveau bloc de donn\'ees)        (2.3)
                        XOR (Ancien bloc de parit\'e)
```

Ces quatre accès pour une écriture constituent donc un goulot d'étranglement. L'objectif de parallèliser les accès à des petites quantités d'informations n'est que partiellement rempli à cause de ces écritures. Ce niveau n'est donc pas à conseiller pour les systèmes transactionnels, et les domaines d'applications de ce niveau restent identiques à ceux du niveau trois. Ce problème de goulot d'étranglement sera réglé dans le niveau cinq.

2.6 Le niveau cinq des disques RAID

Le niveau cinq résoud donc le problème du goulot d'étranglement laissé par le niveau quatre. Pour supprimer ce goulot d'étranglement, ce niveau distribue tous les blocs de parité sur l'ensemble des disques. Il n'y a donc plus de disques contenant uniquement, soit des blocs de données, soit des blocs de parité. Un exemple de répartition des blocs de parités sur un tableau de disques est donné dans la figure 2.5. Toutefois, il faut préciser qu'il existe plusieurs types de répartitions des blocs de parités.

Ces différents placements des blocs de parité seront détaillés dans le chapitre trois car nous en profiterons pour observer les impacts qu'ont ces placements sur les performances.

Au niveau des performances, il faut signaler que ce niveau posséde des performances se rapprochant du niveau un avec des capacités de stockage bien supérieures. Aussi, ce niveau est particulièrement adapté si l'on veut :

- utiliser uniquement des machines massivement parallèles,
- utiliser des systèmes transactionnels en possédant des capacités de stockage faibles,
- utiliser les disques, à la fois avec des systèmes transactionnels, et à la fois avec des machines massivement parallèles.

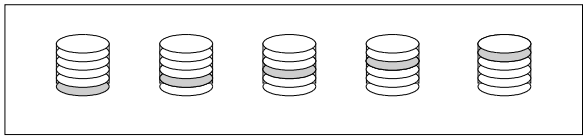

FIG. 2.5 – *Un tableau de disques de niveau cinq*

2.7 Le niveau six des disques RAID

Le niveau six est le dernier niveau, celui-ci est une variante du niveau cinq, la description de ce niveau peut être trouvée dans [13] et [5] mais pas dans les premiers articles de Patterson.

Cette architecture de RAID utilise un code Reed Solomon[4]. L'utilisation d'un code Reed Solomon à la place de blocs de parité permet d'augmenter encore plus la tolérance aux pannes des disques. Ce niveau plus onéreux en disque de redondance est destiné à des applications qui peuvent être utilisées avec des RAID cinq mais qui nécessiteraient un niveau de fiabilité plus élevé. Nous verrons dans le chapitre quatre que ce type de code peut être utilisé avec des disques de secours en ligne.

4. Ce type de code est aussi utilisé en réseau, en particulier sur l'ATM où il permet de corriger des cellules perdues. L'utilisation du code P+Q est conseillée dans ce réseau par l'UIT (ancien CCITT) avec P=124 et Q=4.

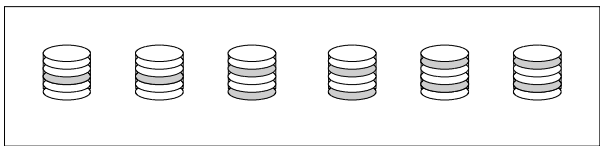

FIG. 2.6 – *Un tableau de disques de type P+Q*

Chapitre 3

Etude des performances des disques RAID

Dans ce chapitre, nous allons aborder un sujet qui a fait l'objet de nombreuses publications sur les disques RAID : l'étude des performances. Bien sûr, on pourrait aborder de nombreux aspects dans ce chapitre, c'est impossible dans ces quelques pages. Aussi, nous avons choisi deux aspects qui nous ont semblés intéressants. Dans le premier paragraphe nous essaierons de déterminer les critères qui permettent d'évaluer les performances. Avec ceci, nous discuterons de l'utilisation qui peut en être faite. Enfin, nous regarderons les impacts sur les performances que peut avoir le placement des blocs de parité par rapport aux blocs de données sur les disques.

3.1 Les critères de performances

Avec l'utilisation de plus en plus fréquente des disques RAID dans l'industrie, il émerge un problème important. En effet, les disques RAID avaient été conçus pour une raison : augmenter les performances des mémoires secondaires. Or justement, C'est sur cette propriété que réside les plus grands problèmes. En effet, on a encore les plus grandes difficultés à évaluer ces performances alors que cette évaluation est nécessaire pour la conception des disques RAID, **mais aussi pour la configuration sur les sites de productions.** Cette configuration est d'autant plus importante que les produits qui arrivent sur le marché aujourd'hui sont très souples : ils permettent entre autre de choisir l'architecture

RAID (cf. l'auto-raid de Hewlett-Packard qui peut fonctionner comme un RAID de niveau un à cinq).

Ces difficultés ne viennent pas tellement des critères de performances eux même. En effet, on recherche en général seulement deux critères qui sont :

- le débit, qui est généralement exprimé en méga octet par seconde, en nombre de requêtes par seconde ou par un pourcentage (par rapport à la charge ou au taux de concurrence[1]),
- le temps de réponse des requêtes d'écriture ou de lecture.

En fait, la difficulté vient du nombre important de paramètres qu'il faut prendre en compte lorsque l'on décide de faire une simulation. Ce qui oblige les utilisateurs à bien connaitre l'utilisation qui va être fait du disque RAID. On peut citer les informations suivantes :

- les niveaux que peut supporter le disque RAID,
- le nombre de disques du RAID,
- le nombre de blocs par disque,
- la taille des blocs (si les blocs sont des cylindres, des pistes ou des secteurs),
- le type des requêtes (si ce sont des requêtes d'écriture ou de lecture, voir un mélange des deux),
- la taille des requêtes (quantité d'information demandée fixe ou variable),
- le degré de concurrence (c'est à dire le nombre de requêtes envoyées simultanément),
- le placement des blocs de parité,
- ect.

Tous ces paramètres influent sur les performances, pire encore, la combinaison de ceux-ci augmente la compléxité du problème. De nombreux auteurs ont effectué des mesures sur des architectures RAID. On peut citer par exemple [2, 3, 6]. En général, pour simplifier le problème, tous les paramètres d'entrées que nous venons d'énumérer ne sont pas pris en compte.

1. par taux de concurrence on entend le nombre de requêtes exécutées en parallèle.

Enfin, pour finir cette présentation des difficultés que représentation l'évaluation des performances des RAIDS, il faut préciser que de nombreux auteurs ont conçu des modèles pour faciliter ces analyses. On peut citer le modèle analytique décrit dans [16], cette référence contient plusieurs références sur d'autres modèles élaborés par d'autres auteurs.

3.2 L'influence du placement des blocs de parité sur les performances

Il existe en fait de très nombreuses possiblités de positionner les blocs de parité par rapport aux blocs de données. Toutefois, toutes ces possiblités ne sont pas valides : en effet, d'après [17, 15, 12], les placements doivent vérifier les deux conditions suivantes :

1. Le placement doit absolument respecter la régle d' "orthogonalité" que nous avons énoncée dans le chapitre quatre : on ne doit pas retrouver sur un même disque deux blocs couverts par un même bloc de parité. On ne doit pas non plus retrouver sur un même disque un bloc de données Q et un bloc de parité Pi si Pi protége Q.
2. de plus, l'équation $j \; div \; n = i$ doit être vérifiée pour tous les blocs de données, avec :
   - n est le nombre de blocs de données protégé par un même bloc de parité.
   - j est le numéro du bloc de données
   - i est le numéro du groupe de parité (par groupe de parité on entend l'ensemble des blocs de données protégé par un même bloc de parité)

Dans les références[17, 15], seulement huit placements sont étudiés. Avant de voir leurs utilisations, nous allons commencer par décrire leur construction. Dans les figures 3.1 à 3.8, les colonnes représentent des disques, les blocs numérotés sont des blocs de données et les blocs en gris et marqués d'un "Pi" sont des blocs de parité. Enfin, les groupes de parité sont constitués par une numérotation séquentielle des blocs de données. Ainsi les blocs de données 0,1,2 et 3 forment un groupe de parité protégé par le bloc P0, de même, on a :

- le bloc P1 protége les blocs 4,5,6 et 7

– le bloc P2 protége les blocs 8,9,10 et 11

– etc .

3.2.1 Placement de type "RAID de niveau zéro"

On ne peut pas vraiment parler de placement de parité sur cette architecture puisqu'il n'y a pas de bloc de parité : ce niveau est cité ici car la construction des placements suivant sera exprimée par le RAID de niveau zéro.

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24

FIG. 3.1 – *Placement du RAID de niveau zéro*

3.2.2 Placement de type "RAID de niveau quatre"

Il est construit à partir du niveau zéro. On ajoute un disque de parité.

0	1	2	3	P0
4	5	6	7	P1
8	9	10	11	P2
12	13	14	15	P3
16	17	18	19	P4

FIG. 3.2 – *Placement du RAID niveau de quatre*

3.2.3 Placement asymétrique à droite

Tous les placements qui suivent sont pour des disques de niveau cinq. Le placement asymétrique à droite est construit à partir du RAID de niveau zéro. On pousse les blocs de données vers la droite à partir de l'endroit où l'on insère le bloc de parité.

P0	0	1	2	3
4	P1	5	6	7
8	9	P2	10	11
12	13	14	P3	15
16	17	18	19	P4

FIG. 3.3 – *Placement asymétrique à droite*

3.2.4 Placement symétrique à droite

Ce placement est construit à partir du RAID de niveau quatre. On effectue un décalage du groupe de parité en entier vers la droite. Le bloc de parité Pi est inséré dans la ligne i, de plus, la rotation se fait sur la même ligne (ainsi pour P1, le bloc sept se retrouve en ligne un, colonne un et non en ligne deux, colonne un).

P0	0	1	2	3
7	P1	4	5	6
10	11	P2	8	9
13	14	15	P3	12
16	17	18	19	P4

FIG. 3.4 – *Placement symétrique à droite*

3.2.5 Placement asymétrique à gauche

Ce placement est construit à partir du RAID de niveau zéro. Ici aussi on pousse les blocs de données vers la gauche à partir de l'endroit où l'on insère le bloc de parité.

0	1	2	3	P0
4	5	6	P1	7
8	9	P2	10	11
12	P3	13	14	15
P4	16	17	18	19

FIG. 3.5 – *Placement asymétrique à gauche*

3.2.6 Placement symétrique à gauche

Ce placement est construit à partir du RAID de niveau quatre. Pour ce faire, on effectue un décalage du groupe de parité en entier vers la gauche. Le bloc de parité Pi est inséré dans la colonne i, de plus, la rotation se fait sur la même ligne (ainsi pour P1, le bloc quatre se retrouve en ligne un, colonne quatre et non en ligne deux, colonne quatre).

0	1	2	3	P0
5	6	7	P1	4
10	11	P2	8	9
15	P3	12	13	14
P4	16	17	18	19

FIG. 3.6 – *Placement symétrique à gauche*

3.2.7 Placement symétrique à gauche étendu

Ce placement est construit à partir du RAID de niveau zéro. On pousse les blocs de données verticalement vers le bas à chaque insertion d'un bloc de parité. On décale à chaque fois l'insertion des blocs de parité vers la gauche.

0	1	2	3	P0
10	11	P2	13	4
P4	21	12	23	14
20	31	22	P6	24
30	P8	32	33	34

5	6	7	P1	9
15	P3	17	8	19
25	16	27	18	P5
35	26	P7	28	29
P9	36	37	38	39

FIG. 3.7 – *Placement symétrique à gauche étendu*

3.2.8 Placement horizontal symétrique à gauche

Ce placement est dérivé du placement symétrique à gauche étendu. La différence avec ce dernier est que le placement horizontal regroupe tous les blocs de parité au même endroit sur tous les disques.

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
P4	P3	P2	P1	P0

FIG. 3.8 – *Placement horizontal symétrique à gauche*

3.2.9 Comparaison des performances

Avant toute comparaison entre ces différents placements, il faut préciser que les impacts des placements sont importants quand la taille

des blocs de données est très importante. Quand celle-ci est très petite, le placement des blocs de parité n'influe presque plus sur les performances. Toutefois, quand la taille des blocs est suffisament importante, le choix du bon placement permet de gagner de 20 à 30 % en débit. La comparaison qui suit est le résultat d'une simulation effectuée par M. Lee dans [15]. Dans cet article, on peut retrouver en page six les différents graphiques qui illustrent ce que nous allons exprimer. M. Lee a étudié les performances des huits placements dans quatre cas :

- en faible charge et avec des accès en lecture : dans ce cas, le classement est fait grâce aux paramètres de distance minimale et distance moyenne[2], ainsi on obtient du plus important au plus faible débit:
  1. le RAID de niveau zéro, les placements symétriques à gauche étendus et horizontaux symétriques à gauche,
  2. les placements symétriques à gauche,
  3. le RAID de niveau quatre, les placements asymétriques à droite, asymétriques à gauche,
  4. les placements symétriques à droite.

- en faible charge et avec des accès en écriture : le RAID de niveau zéro n'ayant pas de blocs de parité, on obtient :
  1. le RAID de niveau zéro,
  2. le RAID de niveau quatre, les placements asymétriques à droite, asymétriques à gauche, symétriques à droite et symétriques à gauche,
  3. les placements symétriques à gauche étendus,
  4. les placements horizontaux symétriques à gauche.

- en forte charge avec des accès en lecture : c'est le RAID de niveau quatre qui est le moins efficace car il ne distribue pas complètement les blocs de données et de parité. Les autres placements ont une efficacité proche les uns des autres.

2. Le paramètre de distance minimale est l'évaluation du minimum des soustractions entre deux numéros de blocs de données appartenants au même disque et à des groupes de parité voisins, ce paramètre permet d'estimer un degré de parallélisme des accès sur les disques (voir[15] en page sept).

- en forte charge avec des accès en écriture : le RAID de niveau zéro est le plus efficace puisqu'il n'a pas besoin d'écrire des blocs de parité. A l'autre extrême, le RAID de niveau quatre est le moins efficace car les blocs de parité constituent un goulot d'étranglement.

Les meilleurs placements pour les RAID de niveau cinq sont donc le symétrique à gauche, le symétrique à gauche étendu et l'horizontal symétrique à gauche. En conclusion de cette comparaison, on peut dire que pour bien optimiser les débits, il faut connaitre le type d'application qui utilisera le disque RAID car les placements avec les plus faibles performances en lecture ont les plus fortes performances en écriture [3] et les placements ayant les plus fortes performances en lecture ont les plus faibles performances en écriture [4].

3. C'est le placement symétrique à gauche.
4. C'est le placement horizontal symétrique à gauche.

Chapitre 4

Les problèmes de tolérance aux pannes

Après avoir étudié les performances des disques RAID, nous allons regarder une propriété importante de ceux-ci : leur résistance aux fautes. En effet, c'est un problème qui n'a pas été abordé dans ce document et qui est primordial dans ces périphériques. Aussi, dans ce chapitre, nous commencerons par poser le problème en donnant quelques définitions, puis nous décrirons le fonctionnement des disques orthogonaux. Nous expliquerons ensuite l'utilité des disques de secours "en ligne". Enfin, nous donnerons des exemples d'algorithmes de reconstructions de parité après une défaillance d'un disque.

4.1 Le problème de fiabilité posé par les disques RAID

Pour augmenter les performances, les concepteurs des disques RAID ont augmenté le nombre de disques, or ceci augmente fortement la probabilité de défaillance du disque RAID : en effet, plus il y a de disques dans le RAID et plus on a de chance qu'un des disques tombe en panne. Cette augmentation de la probabilité de panne a entrainé l'utilisation des blocs de parité que nous avons vue dans le chapitre deux. Cet abaissement de la fiabilité a été constaté dès le début des travaux sur les RAID, ainsi, Patterson dans [21] énonça que :

$$MTTF_{raid} = \frac{MTTF_{disque}}{\text{Nombre de disques dans le RAID}} \tag{4.1}$$

avec :

- $MTTF_{disque}$ est le MTTF d'un disque seul,
- $MTTF_{raid}$ est le MTTF du RAID entier (RAID sans redondance).

REMARQUE :
En matière de fiabilité, les constructeurs ont l'habitude d'exprimer la résistance aux pannes de leurs produits par un paramêtre : le MTTF. Ce paramètre exprime le temps moyen avant la panne (MTTF pour Mean Time To Failure). On parle aussi assez souvent de trois autres paramètres qui sont :

- Le MTTR (ou Mean Time To Repair, c'est à dire le temps moyen de réparation),
- Le MTTDL (ou Mean Time To Data Loss, c'est à dire le temps moyen avant perte d'informations),
- Le MTBF (pour Mean Time Between Failure qui est le temps moyen entre deux pannes).

De la même manière que pour la formule 4.1, Patterson a énoncé la formule donnant le MTTF d'un disque RAID supportant une erreur par groupe de disques[1] :

$$MTTF_{raid} = \frac{(MTTF_{disque})^2}{n_G \cdot G \cdot (G+1) \cdot MTTR_{disque}} \tag{4.2}$$

avec :

- $MTTF_{disque}$ est le MTTF d'un disque seul,
- $MTTF_{raid}$ est le MTTF du RAID entier,
- $MTTR_{disque}$ est le MTTR d'un disque,
- n_G est le nombre de groupes constitués de G disques de données et d'un disque de parité,
- G pour G disques de données,
- $(G+1)$ pour le nombre de disques d'un groupe (G disques de données plus un disque de parité).

1. Par groupe de disques on considére le groupe constitué de G disques plus le disque de parité qui protége CES G disques.

En dehors des équations 4.1 et 4.2 qui ont été énoncées par Patterson, et qui semblent faire une assez bonne approximation de la fiabilité des RAID (d'après les commentaires fait par certains auteurs d'article sur le sujet), il faut préciser que l'on peut aussi appliquer une approche par des modèles markoviens qui sont plus courant quand on parle de fiabilité en général (on peut consulter [1] à cet effet pour la fiabilité sur des disques RAID ou [24] pour une introduction aux modèles markoviens sur la fiabilité en général).

Ajoutons pour finir que Patterson faisait les hypothèses que les pannes étaient indépendants, qu'elles arrivaient de manière régulière, qu'elles suivaient une loi exponentielle. De plus, il ne prennait pas en compte les pannes de composants tel que les contrôleurs de disques.

4.2 Les disques orthogonaux

Dans le précedent paragraphe, nous avons étudié comment les pannes des disques se produisaient. Pour ceci, nous avons fait l'hypothèse implicite que les pannes étaient indépendantes les unes des autres. Dans la réalité, il existe de nombreux cas où cette hypothèse simplificatrice n'est pas valide :

- D'abord à cause des procédés de fabrication des disques : en effet, on peut très bien imaginer qu'une série de plusieurs disques soit victime du même défaut de fabrication à cause d'un modèle de pièce incluse dans chaque disque qui serait lui même défaillant. Il est à noter que ce type de défaut peut conduire à une panne au début de l'utilisation ou vers le MTTF des disques. Un nombre important des disques tombent alors en panne quasiment en même temps : leurs pannes ne sont plus indépendantes les unes des autres,
- ou à cause de la défaillance d'un composant du RAID qui est utilisé par plusieurs disques. C'est le cas d'une alimentation, d'un contrôleur ou d'un ventilateur qui peut être utilisé par plusieurs disques.

Un exemple de structure orthogonale est donné dans la figure 4.1. Dans cette figure, un composant (représenté par un rectangle) peut, par exemple, être un contrôleur. Chaque ligne de disque correspond à un

RAID avec son disque de parité (représenté avec un P sur le disque[2]). Ainsi, si un disque d'une ligne tombe en panne, on peut récupérer les données comme à l'habitude. Si maintenant, c'est un contrôleur qui tombe en panne, on voit que c'est tout un groupe (dit "groupe par composant") qui ne fonctionne plus. Toutes les lignes sont alors solicitées pour la reconstruction d'un disque par ligne. La charge de reconstruction est alors très forte mais on n'a pas perdu de données. Si les "groupes par composant" correspondaient aux "groupes par parité", dans ce cas on aurait perdu définitivement les données de la ligne de disques en panne.

En conclusion de ce paragraphe, on voit bien que l'on ne peut pas se permettre comme l'a fait Patterson, d'ignorer dans le calcul du $MTTF_{raid}$ les composants tels que alimentation, contrôleur ou ventilateur. Cette constatation a été démontrée par M. Schulze dans [23]. Dans cet article, les MTTF de chaque composant sont pris en compte dans le calcul du MTTF du disque RAID. La formule de Patterson devient alors :

$$MTTF_{raid} = \frac{(MTTF_{disque})^2}{n_G \cdot G \cdot (G+1) \cdot MTTR_{disque} \cdot (1 + \alpha_F \cdot \frac{1+\alpha_R}{\alpha_R} + \frac{\alpha_F^2}{n_G \cdot \alpha_R})} \tag{4.3}$$

pour $\alpha_F = \frac{MTTF_{disque}}{MTTF_{colonne}}$ et $\alpha_R = \frac{MTTR_{disque}}{MTTR_{colonne}}$

avec :

- $MTTR_{disque}$ est le temps moyen de réparation d'un disque
- $MTTR_{colonne}$ est le temps moyen de réparation d'un composant

Un exemple d'application numérique des formules 4.1, 4.2 et 4.3 peut être trouvé dans l'article [23].

Bien que cette architecture permette d'augmenter sensiblement la résistance aux pannes des mémoires secondaires, celle-ci ne permet pas de satisfaire la contrainte de forte disponibilité que nécessite certaines applications. Cette propriété sera étudiée dans le paragraphe suivant.

2. Ici on considère que les blocs de parité sont centralisés sur un seul disque. Toutefois, s'ils étaient distribués sur la ligne (comme des RAID de niveau cinq), ceci n'aurait aucun effet sur le fonctionnement du RAID orthogonal. Cette simplification est seulement faite pour aider à la compréhension des propriétés des RAID orthogonaux.

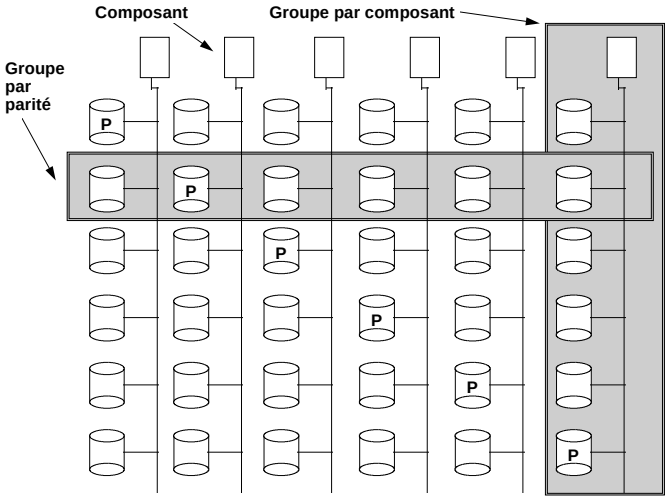

FIG. 4.1 – *Les disques RAID orthogonaux*

4.3 Les disques de secours "en ligne" (ou "standby spare disks")

Jusqu'à présent, nous n'avons pas parlé de ce qu'il se passait après une panne. En effet, après une panne, les données contenues dans le disque perdu doivent être recalculées par des "ou exclusifs" entre les données qui restent accessibles. **Durant cette période, une deuxième panne entraŇne une perte d'informations**, de plus, la reconstruction entraine une forte charge supplémentaire sur le RAID, abaissant ainsi les performances. **L'objectif de la phase de reconstruction est alors de recalculer les données perdues au plus vite.** Deux facteurs peuvent permettre d'atteindre cet objectif : l'utilisation d'algorithmes de reconstruction efficaces (examinés dans le paragraphe suivant) et la disponibilité de disques en ligne que nous allons voir tout de suite.

Un disque en ligne est en fait un disque ajouté à un RAID et ne stockant aucune information quand le RAID fonctionne normalement. Son utilité est de réduire le paramètre MTTR des disques : en effet, lors de la panne d'un disque, le RAID ne devra pas attendre une intervention humaine

pour le remplacement du disque endommagé avant de commencer la reconstruction des données, il utilisera directement le disque en ligne et exécutera la reconstruction immédiatement. Le disque en panne pourra alors être changé ultérieurement (on peut imaginer qu'un technicien regarde une fois par jour les disques en panne et les change à ce moment, dans ce cas, les disques en ligne permettent aussi d'assouplir les actions de maintenance sur les mémoires secondaires).

Il existe plusieurs type de disques en ligne, en effet, de nombreux auteurs ont essayé d'optimiser ce mécanisme, car durant la phase normale de fonctionnement, le disque en ligne ne fonctionne pas du tout et reste totalement inactif. On trouve des exemples d'optimisation dans [18] et [5]. D'après [5], il existe deux types d'optimisations :

- les disques en ligne avec blocs[3] de secours distribués, les blocs de secours n'étant pas utilisés (approche dite "distributed sparing"),
- les disques en ligne avec blocs de secours distribués et utilisés pour stocker des informations de parité et améliorer la fiabilité (approche dite "parity sparing").

Quant à [18], il fait aussi une distinction entre disques en ligne partagés entre plusieurs RAID et disques en ligne alloués par un seul RAID[4]. On obtient alors les quatre types de disque en ligne suivants :

- Les disques en ligne avec blocs de secours centralisés représentés par la figure 4.2 (sur les quatre figures suivantes, les blocs de secours sont marqués par le libellé "Si", les blocs de parité par "Pi" et les blocs de données par des numéros).
- Les disques en ligne avec blocs de secours distribués sur tous les disques. Les blocs de secours n'étant pas partagés entre plusieurs RAID (un bloc de secours ne pouvant pas être utilisé que par un seul RAID). Cette topologie est représentée par la figure 4.3.
- Les disques en ligne avec blocs de secours distribués sur tous les disques ou centralisés (en fait, les deux types précédents), mais les

3. La notion de blocs de secours est la même que celle des blocs de parité : les blocs sont des secteurs, des cylindres ou des pistes.

4. **Par partagé, on entend que par exemple, si dans un système on a un disque en ligne disponible et plusieurs RAID, n'importe quel RAID ayant une panne peut utiliser ce disque disponible.**

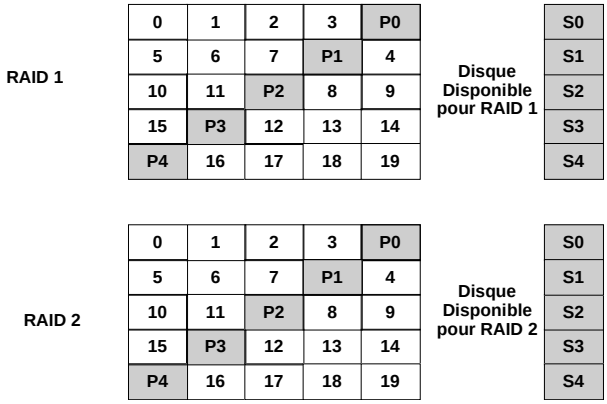

FIG. 4.2 – *RAID avec un disque dédié de type "en ligne" partagé ou non*

blocs de secours étant partagés entre plusieurs RAID. Ces topologies sont aussi représentées par les figures 4.3 et 4.2.

– Les disques en ligne avec des blocs de secours distribués dans tout le RAID et qui sont utilisés pour stocker des informations de parité (voir la figure 4.4, on peut ainsi mettre en oeuvre un RAID de niveau P+Q, puis passer en un RAID de niveau cinq après une panne).

Même si toutes ces optimisations permettent non seulement d'utiliser les disques en ligne afin de partager la charge de travail sur un nombre supérieur de disques, mais en plus, de permettre de diminuer le temps de reconstruction des disques en panne car ceux-ci sont moins pleins, Il faut toutefois signaler que le fait qu'il y ait plus de disque en fonctionnement augmente aussi la probabilité d'une panne d'un disque !

4.4 Les algorithmes de reconstruction de parité

Après les disques en ligne, il nous reste un dernier aspect de la tolérance aux pannes dans les RAID : les algorithmes de reconstruction.

Nous avons vu qu'une fois que l'on a détecté la panne d'un disque, le RAID doit déclencher des algorithmes de reconstruction pour récu-

RAID 1

0	1	2	3	P0	S0
5	6	7	P1	S1	4
10	11	P2	S2	8	9
15	P3	S3	12	13	14
P4	S4	16	17	18	19

RAID 2

0	1	2	3	P0	S0
5	6	7	P1	S1	4
10	11	P2	S2	8	9
15	P3	S3	12	13	14
P4	S4	16	17	18	19

FIG. 4.3 – *RAID avec blocs de secours distribués partagés ou non*

pérer les données perdues. Ces algorithmes sont déclenchés dès qu'ils disposent d'un disque sur lequel ils peuvent réécrire les données. Nous avons aussi vu plus haut qu'il est très important de diminuer au maximum ce temps de reconstruction. Malgrè tout, la diminution du temps de reconstruction n'est pas le seul critère de performances qui rentre en ligne de compte dans le choix de l'algorithme de reconstruction : certaines applications nécessitent une grande disponibilité des données. Or, comme on l'a déjà dit, la phase de reconstruction va augmenter la charge sur le RAID et risque donc d'augmenter les temps de réponse des accès disques par les utilisateurs. Ce deuxième critère doit parfois être pris en compte, et bien sûr optimisé.

Ce sujet a fait l'objet de nombreuses publications. Et les auteurs ont déterminé plusieurs facteurs qui interagissent sur nos deux critères de performances. Ce sont par exemple :

- L'architecture du RAID : en effet, les performances de la reconstruction sont différentes entre un disque RAID de niveau cinq et un disque miroir. Pour une architecture de type miroir, quand un disque tombe en panne, le disque miroir reĞoit tous les accès utilisateurs qui étaient auparavant répartis sur les deux disques (plus les accès pour la reconstruction !). L'augmentation de la charge est nettement moins importante pour les RAID de niveau cinq,
- la taille des blocs dans les disques : selon que les blocs soient

RAID 1

0	1	2	3	P0	P0 bis
5	6	7	P1	P1 bis	4
10	11	P2	P2 bis	8	9
15	P3	P3 bis	12	13	14
P4	P4 bis	16	17	18	19

RAID 2

0	1	2	3	P0	P0 bis
5	6	7	P1	P1 bis	4
10	11	P2	P2 bis	8	9
15	P3	P3 bis	12	13	14
P4	P4 bis	16	17	18	19

FIG. 4.4 – *RAID avec des blocs de secours utilisés comme blocs de parité*

des pistes, cylindres ou secteurs, les performances sont différentes (dans [10] on peut trouver une étude sur ceci),

- les priorités que l'on donne aux accès utilisateurs et aux accès pour la reconstruction : la priorité aux accès utilisateurs devra être plus forte si l'on veut privilégier les temps de réponse par rapport au temps de reconstruction, et inversement,
- l'algorithme de reconstruction : il en existe beaucoup. Dans [10] on trouve des références bibliographiques sur bon nombre d'entre eux. L'algorithme donné dans l'exemple ci dessous est très simple. Il peut être amélioré par une parallélisation : on peut créer un processus par bloc à reconstruire. Certains algorithmes plus performants peuvent aussi tirer partie de l'architecture du RAID, on trouve par exemple dans [9] un algorithme adapté aux disques avec parité "dégroupée" (en anglais : "declustered parity disks"). Cet algorithme permet de mieux profiter du débit potentiel du RAID non utilisé par les accès utilisateurs. Son principe est que chaque disque "vivant"[5] envoie son bloc vers une mémoire centrale ou sera effectué le XOR, puis d'où sera envoyé le résultat final vers le disque remplaçant celui en panne,

```
Pour Chaque bloc du disque en panne Faire
    1) Identifier a quel groupe de parite
```

5. on entend par là : tous les disques du RAID qui ne sont pas en panne.

```
            appartient le bloc,
        2) Envoyer des requetes de lecture de
            tous les autres blocs encore "vivants"
            du groupe de parite. Les requetes
            ont une priorite faible pour limiter
            l'effet sur les temps de reponse pour
            les utilisateurs
        3) Attendre tous les blocs demandes
        4) Effectuer le XOR entre tous ces blocs
        5) Ecrire le nouveau bloc avec une requete
            de priorite faible
    Refaire
```

- enfin, certaines techniques peuvent être utilisée (comme la reconstruction "au-vol" : quand un utilisateur a besoin d'un bloc du disque défaillant, il n'attend pas la fin de la reconstruction. En effet, on souhaite pénaliser le moins possible son temps de réponse. Aussi, on reconstruit le bloc désiré et on le lui fournit. La reconstrution "au-vol" en profite alors pour écrire le bloc délivré sur le nouveau disque en place, ce qui lui économise le calcul d'un bloc. Cette technique est aussi appelée "piggybacking" par certains auteurs. Il faut remarquer que les accès utilisateurs aux blocs du disque en panne accentuent aussi la charge du RAID.

Chapitre 5

Exemples de prototypes ou de produits industriels

Il existe aujourd'hui un bon nombre d'exemples de disques RAID. Nous étudierons deux prototypes qui correspondent, au premier prototype de disque RAID de Berkeley (qui est le RAID I, référencé dans [20, 12]), et au deuxième prototype de Berkeley (qui est RAID II, référencé dans [8, 14]). Enfin, nous donnerons quelques références d'autres produits, sans pour autant les décrire précisément.

5.1 Le prototype RAID I

Le prototype RAID I fut le premier RAID construit. Il a été réalisé par l'équipe RAID de Berkeley (plus précisément par [14]). Du point de vue matériel, il est constitué d'une machine Sun 4/280 avec cent vingt huit méga octet de mémoire centrale, d'interfaces SCSI[1] et de trente deux disques de cinq pouces un quart (en fait, il y a huit interfaces SCSI avec, pour chaque interface quatre disques).

Ce prototype est un RAID de niveau cinq. Il a été conçu pour fonctionner sur le système d'exploitation Sprite. Bien que ses performances ne furent pas complètement satisfaisante, il a permi de mesurer les performances d'un disque RAID et d'introduire les placements de parité que nous avons étudiés dans le chapitre trois. Sa grande réussite est que vue du système d'exploitation, l'ensemble des disques apparaissait

1. SCSI pour Small Computer System Interface.

comme un disque unique : tous les traitements RAID étaient réalisés par un pilote de périphérique développé à cette occasion.

En fait, RAID I apporta de bonnes performances pour des petits accès disques de type "transactionnels", mais Lee démontra qu'il n'était pas adapté pour les gros débits nécessaires aux machines massivement parallèles (avec RAID I, Lee a obtenu des débits de l'ordre de 2,3 méga octet par seconde alors que chaque disque permettait des débits de l'ordre de 1,3 méga octet par seconde). Les raisons essentielles de ce problème étaient :

- la mémoire de la station Sun était rapidement saturée,
- le processeur qui n'était pas adapté pour de gros débits d'entrées/sorties car chaque entrée/sortie transitait par un cache du processeur adressé par des adresses virtuelles.
- et enfin car le débit du bus VME de quarante méga octet par seconde était en fait limité à neuf méga octet par seconde (Le bus était très rapidement saturé).

L'objectif de Lee étant de concevoir un périphérique capable de fonctionner avec ces deux types d'accès, le deuxième prototype, RAID II, fut réalisé.

5.2 Le prototype RAID II

Comme nous l'avons dit plus haut, les objectifs de RAID II étaient ambitieux. RAID II devait permettre de faire face à :

- d'importants débits pour des architectures massivement parallèles,
- de nombreux accès simultanés pour les applications transactionnelles,
- de nouvelles applications qui sont elles aussi très gourmandes en débit (comme la CAO[2], le multimédia ou les bases de données).

Pour atteindre ces objectifs, RAID II a gagné en complexité et a été construit pour s'exécuter sur Sprite LFS (qui est un système de fichiers optimisé pour permettre de gros débits d'entrées/sorties). La figure 5.1 décrit l'architecture de ce prototype.

2. CAO pour Conception assistée par ordinateur

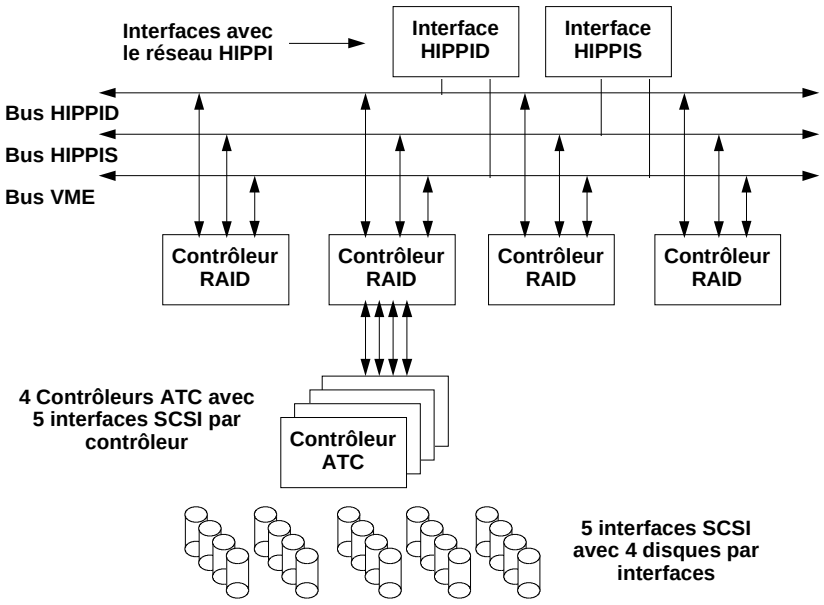

FIG. 5.1 – *Le prototype RAID II*

En fait, RAID II n'est pas un périphérique que l'on branche à une machine. En effet, pour atteindre de gros débits, RAID II est connecté à un réseau HIPPI[3]. Cette connexion se fait par deux interfaces (interface HIPPID et interface HIPPIS sur la figure 5.1). Ensuite, on trouve trois bus, deux bus de la société "Thinking Machine Corporation" et un bus VME. En fait, les deux bus HIPPI transportaient les données et le bus VME véhiculait les commandes. Ce découpage en deux supports était justifié pour des problèmes de performances. C'est sur ces bus qu'étaient connectés les unités RAID.

Ces unités sont en fait quasiment les seules pièces réalisées par le groupe RAID de Berkeley, tous les autres composants ont été achetés chez IBM, Thinking Machine Corporation, Interphase Corporation et Sun. Ces unités RAID (contrôleur RAID sur la figure 5.1) sont en fait

3. HIPPI pour HIgh Performance Parallel Interface, ce type de réseau peut atteindre des débits de 1600 méga bits par seconde en simplex et 800 méga bits par seconde en duplex : ce qui explique peut être que RAID II soit connecté à un réseau HIPPI par deux interfaces unidirectionnelles !

constituées de deux entités[4] :

- quatre modules mémoires de 32 méga octet chacun,
- un crossbar appelé XBUS, qui connecte les deux ports HIPPIS et HIPPID, les modules mémoires, une unité chargée de calculer les parités ainsi que les quatre ports VME qui sont connectés à des contrôleurs ATC. Ces contrôleurs ATC fournissent chacun cinq interfaces SCSI permettant de connecter les disques.

Pour finir, comme pour RAID I, des études ont été effectuées pour évaluer ses performances. Celles-ci sont bien meilleures que RAID I : en lecture, alors que RAID I permettait un débit de 2,4 méga octet par seconde, RAID II offre un débit de 20,9 méga octet. En écriture on obtient un débit de 18,2 méga octet alors qu'il était de 1,2 méga octet pour RAID I. Bien sûr, de nombreux facteurs influent sur ces résultats, une étude plus détaillée des performances sur ce prototype pourra être consultée dans [4].

5.3 Références d'autres produits

Il existe en fait d'innombrable prototypes et produits industriels. On peut citer par exemple :

- Un système de gestion de base de données a été conçu avec des disques RAID afin d'offrir des débits importants et un degré de disponibilité élevé. Celui ci est une adaptation du SGBD[5] Postgres : il s'appelle XPRS[6] et on peut trouver une description de celui-ci dans [25],
- des systèmes de fichiers ont été réalisés, on peut citer Sprite LFS[7] décrit dans [22] ou Zebra (qui est basé sur certaines idées introduites par Sprite LFS,), décrit dans [27]. Enfin, des projets d'adaptation de RAMA[8] sur des disques RAID ont été envisagés dans [19],

4. Une description détaillée de ces contrôleurs peut être consultée dans [8]
5. SGBD pour Système de Gestion de Base de Données.
6. XPRS pour eXtented Postgres on Raid and Sprite.
7. LFS pour Log structured file system. Sprite LFS est le système de fichiers du système d'exploitation Sprite.
8. RAMA pour Rapid Access to Massive Archive.

- citons aussi en vrac et dans le désordre : le TMC Scale Array qui est un RAID niveau trois pour les "Connection Machines", Iceberg de la société StorageTek, le projet de chez Hewlett-Packard : TickerTAIP/DataMesh, qui avait pour objectif de relier très rapidement des sites de stockage avec un réseau fiable, les nouveaux produits RAID S et auto-raids de chez Hewlett-Packard et EMC, le mini ordinateur CONVEX C240 qui était équipé d'un disque RAID niveau 5, la machine du laboratoire national de Los Alamos (RAID de niveau trois contrôlé par un IBM RISC/6000 avec une liaison à base d'un réseau HIPPI), ainsi que les serveurs de fichiers Solbourne 5E/905 (tous ces produits ont été trouvés dans [7, 5, 4]),
- etc.

Chapitre 6

Conclusion et perspectives d'avenir des disques RAID

Dans ce document, nous avons examiné les différentes architectures de disques RAID ainsi que leurs propriétés. Dans l'introduction, nous avions justifié l'utilisation des disques RAID dans les architectures massivement parallèles et dans les systèmes transactionnels par le fait que les processeurs et les mémoires principales évoluaient plus vite que les mémoires secondaires. A notre avis, il y a de forte chance pour que cette évolution continue dans le futur, voir s'accentue ! D'ailleurs, après toutes les études qui ont étés faites par le projet Berkeley et une standardisation par le RAB[1], c'est aujourd'hui l'industrie qui prend le relais. Les constructeurs tentent de faire évoluer les disques RAID pour

- tirer profit des nouvelles interfaces SCSI-2, Ultra-SCSI, et des bus tels que les bus PCI,
- pouvoir équiper tous les types de systèmes (de la micro au grands systèmes),
- permettre de s'attaquer à de nouveaux domaines applicatifs (comme la vidéo),
- permettre de gagner encore plus en performances et en consommation d'énergie.

Certaines annonces comme les disques RAID S de la société EMC (variante de RAID niveau quatre) ou les disques auto-raids de HP (qui

1. RAB pour Raid Advisory Bureau, c'est un ensemble de constructeurs regroupant entre autres Adaptec, Conner, IBM et Digital.

permettent la migration dynamique des données en fonction de leur utilisation sur des volumes configurés dans un mode RAID donné) font donc leur apparition dans les catalogues[11].

De plus, le marché des disques RAID est en pleine expansion : en 1993, il était de 3,5 milliards de dollars, en 1994, il était évalué à 5 milliards de dollars, les prévisions pour 1995 devraient être encore meilleures (si l'on regarde la figure 6.1, on constate aussi que les disques RAID se généralisent à des plateformes de plus en plus variées : 74 % et 66,5 % correspondant aux pourcentages de disques RAID dans les systèmes mini et micro avec réseaux locaux, 24,5 % et 32 % sont les pourcentages pour les grands systèmes, 1% pour les systèmes à hautes performances et 0,5 % pour les systèmes mono utilisateur, source provenant de [11]).

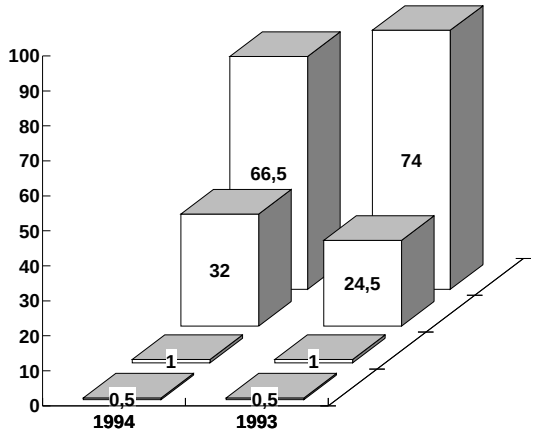

FIG. 6.1 – *Evolution du marché des disques RAID en 1993 et 1994*

En conclusion, on peut dire que les disques RAID, dont le regain d'intérêt fut important ces dernières années, vont certainement encore se développer, et les RAID devraient progressivement devenir des mémoires secondaires courantes dans tous les types de systèmes informatiques modernes.

Bibliographie

[1] W. Burkhard and J. Menon. Disk array Storage System Reliability. pages 432–444. Proc. IEEE, 1993.

[2] P. M. Chen, G. A. Gibson, R. H. Katz, and D. A. Patterson. An Evaluation of Redundant Arrays of Disks Using an Amdahl 5890. Proceedings of the 1990 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, 1990.

[3] P. M. Chen and E. K. Lee. Striping in a RAID Level 5 Disk Array. Technical Report CSE-TR-181-93.

[4] P. M. Chen, E. K. Lee, A. L. Drapeau, K. Lutz, E. L. Miller, S. Seshan, K. Shirriff, D. A. Patterson, and R. H. Katz. Performance and Design Evaluation of the RAID-II Storage Server. International Parallel Processing Symposium Workshop on I/O in Parallel Computer Systems, 1993.

[5] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. RAID : High-Performance, Reliable Secondary Storage. Technical Report UCB/CSD-93-778.

[6] P. M. Chen and D. A. Patterson. Maximizing Performance in a Striped Disk Array. Proc. International Symposium on Computer Architecture, 1990.

[7] P. M. Chen and D. A. Patterson. A New Approach to I/O Performance Evaluation Self-Scaling I/O Benchmarks, Predicted I/O Performance . pages 1–12. Santa Clara, California, Proceedings of the 1993 ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems, 1993.

[8] R. H., Katz, P. M. Chen, A. L. Drapeau, E. K. Lee, K. Lutz, E. L. Miller, S. Seshan, and D. A. Patterson. RAID-II : Design and Implementation of a Large Scale Disk Array Controller. Symposium on Integrated Systems, 1993.

[9] M. Holland, G. Gibson, and D. Siewioreck. Fast, On-line Failure Recovery in Redundant Disk Arrays. pages 140–149. Proc. International Symposium on Fault-Tolerant Computing, 1993.

[10] R. Y. Hou and Y. N. Patt. Comparing Rebuild Algorithms for Mirrored and RAID 5 disk arrays. pages 317–326. University of Michigan, Proc. ACM SIGMOD, 1993.

[11] T. Jacquot. Le 01 informatique numéro 1356 : Les technologies du Raid évoluent tous azimuts. page 4, 1995.

[12] E. K. Lee. Software and Performance Issues in the Implementation of a RAID Prototype. Technical Report UCB/CSD 90/573, University of California at Berkeley, 1990.

[13] E. K. Lee. Performance Modeling and Analysis of Disk Arrays. Technical Report UCB/CSD 93/770, University of California at Berkeley, 1993.

[14] E. K. Lee, P. M. Chen, J. H. Hartman, L. C. Drapeau, E. L. Miller, R. H. Katz, G. A. Gibson, and D. A. Patterson. RAID-II : A Scalable Storage Architecture for High-Bandwidth Network File Service. Technical Report UCB/CSD 92/672, University of California at Berkeley, February 1992.

[15] E. K. Lee and R. H. Katz. Performance Consequences of Parity Placements in Disk Arrays. Proc. SIGMETRICS.

[16] E. K. Lee and R. H. Katz. An Analytic Performance Model of Disk Arrays. Proc. SIGMETRICS, 1993.

[17] E. K. Lee and R. H. Katz. The Performance of Parity Placements in Disk Arrays. *IEEE Trans. on Computers*, 42, 1993.

[18] J. Menon and D. Mattson. Distributed sparing in disk arrays. pages 410–421. Proc. IEEE, 1992.

[19] E. L. Miller and R. H. Katz. RAMA : a filesystem for massively parallel computers. Proc. IEEE Symposium on Mass Storage Systems, 1993.

[20] D. A. Patterson, P. Chen, G. Gibson, and R. H. Katz. Introduction to Redundant Arrays of Inexpensive Disks (RAID). pages 112–117. 1989.

[21] D. A. Patterson, G. Gibson, and R. H. Katz. A Case for Redundant Arrays of Inexpensive Disks (RAID). pages 109–116. 1988.

[22] M. Rosenblum and J. K. Ousterhout. The design and implementation of a log structured file system. University of California, Berkeley, ACM Trans. on Computer Systems, 1992.

[23] M. Schultze, G. Gibson, R. Katz, and D. Patterson. How reliable is a RAID? Proc. IEEE, 1989. COMPCON 89.

[24] Martin L. Shooman. Software engineering. pages 550–600, 1983.

[25] M. Stonebraker, R. Katz, and D. Patterson J. Ousterhout. The design of XPRS. pages 318–330. Proc. Very Large Data Bases, 1988. University of California, Berkeley.

[26] A. Tanenbaum. *Réseaux: architectures, protocoles, applications.* Inter-éditions, 1989.

[27] Zebra:a striped network file system. Proceedings of the USENIX Workshop on File Systems, 1992.