RAID Triple Parity

Григоренко Павел

2024-03-24

1. Принцип построения

Алгоритм RAID Triple Parity (RTP), описанный в [1], является расширением алгоритма Row-Diagonal Parity (RDP), описанного в [2], так что для начала затроним его.

1.1. RDP

За основу берётся RAID-4 массив, состоящий из $p - 1$ дисков с данными, где p — простое число, и одного диска с чек-суммами[1] рядов (далее «R»). К этому массиву добавляется ещё один диск (далее «Diag») с чек-суммами, только считаются они не по рядам (как в R), а «по диагоналям».

Опишем, что это значит, более формально.

Массив разбивается на группы по $p - 1$ рядов. Диагональные чек-суммы в каждой такой группе считаются отдельно, без учёта всех остальных групп. Как следствие, восстановление данных в такой группе также может происходить независимо от остальных групп.

Более точно, если пронумеровать диски с данными от 0 до $p - 2$ включительно, а диску R присвоить номер $p - 1$, то (при нумерации рядов с нуля) блок, находящийся в ряду j на диске i, будет принадлежать диагонали с номером $(i + j) \bmod p$.

Таблица 1. Разбиение на диагонали в RDP при $p = 7$

Диск 0	Диск 1	Диск 2	Диск 3	Диск 4	Диск 5	Диск R
0	1	2	3	4	5	6
1	2	3	4	5	6	0
2	3	4	5	6	0	1
3	4	5	6	0	1	2
4	5	6	0	1	2	3
5	6	0	1	2	3	4

Таким образом мы получаем p диагоналей. Для $p - 1$ из них подсчитывается чек-сумма и записывается в Diag.

Таблица 2. Подсчёт диагональных чек-сумм в RDP при $p = 7$

Диск 0	Диск 1	Диск 2	Диск 3	Диск 4	Диск 5	Диск R	Diag
0	1	2	3	4	5		0
1	2	3	4	5		0	1
2	3	4	5		0	1	2
3	4	5		0	1	2	3
4	5		0	1	2	3	4
5		0	1	2	3	4	5

Стоит отметить, что данные в Diag не учитываются при подсчёте R, но данные R учитываются при подсчёте Diag. То есть исходный RAID-4 массив остаётся неизменённым, и для восстановления данных при отказе одного диска из исходного массива можно использовать стандартную процедуру.

1.2. RTP

Добавим к исходному RAID-4 массиву ещё один диск с диагональными чек-суммами, только в этот раз с другим разбиением на диагонали: используется формула $(i - j - 1) \bmod p$. Условимся называть такие диагонали «анти-диагоналями».

[1] Здесь и далее под «чек-суммами» подразумеваются XOR-суммы

Таблица 3. Разбиение на анти-диагонали в RTP при $p = 7$

Диск 0	Диск 1	Диск 2	Диск 3	Диск 4	Диск 5	Диск R
6	0	1	2	3	4	5
5	6	0	1	2	3	4
4	5	6	0	1	2	3
3	4	5	6	0	1	2
2	3	4	5	6	0	1
1	2	3	4	5	6	0

Получаем p анти-диагоналей, чек-суммы для $p - 1$ из них записываем в новый диск, «A-Diag».

Таблица 4. Подсчёт анти-диагональных чек-сумм в RTP при $p = 7$

Диск 0	Диск 1	Диск 2	Диск 3	Диск 4	Диск 5	Диск R	A-Diag
6		1	2	3	4	5	6
5	6		1	2	3	4	5
4	5	6		1	2	3	4
3	4	5	6		1	2	3
2	3	4	5	6		1	2
1	2	3	4	5	6		1

Стоит отметить, что данные Diag при подсчёте анти-диагональных сумм никак не учитываются.

2. Восстановление стираний

Для удобства введём собирательное название для дисков с данными и диска R: «RAID-4 диски».

2.1. Отказ одного диска

2.1.1. Diag/A-Diag

Переподсчёт чек-сумм.

2.1.2. Один RAID-4 диск

Можно убрать Diag и A-Diag из рассмотрения и получить, по сути, RAID-4 массив, в котором произошёл отказ одного диска. Процедура восстановления очевидна.

2.2. Отказ двух дисков

2.2.1. Diag и A-Diag

Переподсчёт чек-сумм.

2.2.2. Один RAID-4 диск и Diag/A-Diag

Восстанавливаем RAID-4 диск, используя процедуру для RAID-4. Переподсчитываем Diag/A-Diag.

2.2.3. Два RAID-4 диска.

Можно воспользоваться процедурой восстановления для RDP, описанной в [2].

Если кратко, то мы можем найти две диагонали, на которых был стёрт только один блок. Хотя бы для одной из этих диагоналей у нас хранится чек-сумма.[2] С помощью этой чек-суммы можем восстановить тот самый блок. Теперь у нас есть ряд, в котором стёрт только один блок (второй мы только что восстановили).

[2]Вообще, чек-сумму диагонали, которая не была сохранена в Diag, можно восстановить как сумму всех остальных диагональных чек-сумм. Это следует из того, что сумма всех диагональных чек-сумм равна сумме всех блоков на RAID-4 дисках (по построению), что, в свою очередь, равняется нулю т.к. сумма блоков в каждом ряду равняется нулю благодаря наличию R, а сумма нулей — это, вполне ожидаемо, ноль. (Аналогичное рассуждение применимо к анти-диагоналям.)

Воспользуемся чек-суммой ряда, чтобы восстановить и его. Таким образом мы восстановили целый ряд. Снова можем найти две диагонали, на которых был стёрт только один блок.[3] Повторяем эту процедуру $p-1$ раз.

2.3. Отказ трёх дисков

2.3.1. Один RAID-4 диск, Diag и A-Diag.

Восстанавливаем RAID-4 диск, используя процедуру для RAID-4. Переподсчитываем Diag и A-Diag.

2.3.2. Два RAID-4 диска и A-Diag

Используем процедуру для RDP, переподсчитываем A-Diag.

2.3.3. Два RAID-4 диска и Diag

Заметим, что процедуру восстановления для RDP можно использовать и с A-Diag вместо Diag. Так что сначала воспользуемся ей, а затем переподсчитаем Diag.

2.3.4. Три RAID-4 диска

Пронумеруем RAID-4 диски, начиная с нуля (диск R будет иметь номер $p-1$). Также пронумеруем ряды, тоже начиная с нуля.

Добавим в конец ещё один, воображаемый ряд (с номером $p-1$). Блоки RAID-4 дисков в этом ряду будут содержать нули, а Diag и Anti-Diag будут «хранить»[2] чек-суммы для той диагонали и анти-диагонали, чек-суммы которых изначально не были записаны на диски.

Пусть отказали диски X, Y и Z. Тогда для $\forall k \in \{0, ..., p-1\}$ можем записать линейное уравнение

$$Y_k \oplus Y_{k+Z-Y} \oplus Y_{k+Y-X} \oplus Y_{k+Z-X} = \mathrm{XOR}_r(k) \oplus \mathrm{XOR}_d(Z,k) \oplus \mathrm{XOR}_a(X,k) \oplus \mathrm{XOR}_r(q)$$

Y_j — блок в ряду $j \bmod p$ на диске Y,
$\mathrm{XOR}_r(j)$ — чек-сумма стёртых блоков в ряду j,
$\mathrm{XOR}_d(i,j)$ — чек-сумма стёртых блоков на диагонали, содержащей блок в ряду j на диске i,
$\mathrm{XOR}_a(i,j)$ — чек-сумма стёртых блоков на анти-диагонали, содержащей блок в ряду j на диске i,
q — ряд, в котором диагональ блока $A[Z,k]$ пересекает диск X (а именно $(k+Z-X) \bmod p$).

Итого имеем систему из p линейных уравнений относительно p переменных (одна из которых заведомо равна нулю). Восстановим диск Y, решив эту систему.

Оставшиеся два диска восстановим с помощью RDP (см. Раздел 2.2.3).

Библиография

[1] A. Goel и P. Corbett, «RAID triple parity», *SIGOPS Oper. Syst. Rev.*, т. 46, вып. 3, сс. 41–49, дек. 2012, doi: 10.1145/2421648.2421655.

[2] P. Corbett *и др.*, «Row-diagonal parity for double disk failure correction», в *Proceedings of the 3rd USENIX Conference on File and Storage Technologies*, в FAST'04. San Francisco, CA: USENIX Association, 2004, с. 1–2.

[3] Чтобы это было так, необходимо, чтобы p было простым.