Xingjun ZHANG, Ningjing LIANG, Yunfei LIU, Changjiang ZHANG, Yang LI, 2022. SA-RSR: a read-optimal data recovery strategy for XOR-coded distributed storage systems. *Frontiers of Information Technology & Electronic Engineering*, 23(6):858-875. https://doi.org/10.1631/FITEE.2100242

SA-RSR: a read-optimal data recovery strategy for XOR-coded distributed storage systems

Key words: Distributed storage system; Data reliability and availability; XOR-based erasure codes; Single-node failure; Data recovery

Corresponding author: Xingjun ZHANG
E-mail: xjzhang@xjtu.edu.cn
ORCID: https://orcid.org/0000-0003-1434-7016

Motivation

1. Minimizing the number of symbols to be read for single data-node failure of XOR-based erasure codes is an NP-hard problem, and existing enumeration recovery has extremely high time complexity.

2. Hill-climbing recovery can obtain a suboptimal symbol-reading scheme in polynomial time.

3. We propose a random search recovery algorithm based on simulated annealing (SA). The algorithm has low time complexity and overcomes the problem of easily being trapped in local optima in the search process.

Method

Our random search recovery algorithm SA-RSR can be decomposed into two steps:

(1) parity symbol grouping;

(2) recovery solution search.

Method

Step 1: parity symbol grouping

Algorithm 1 Parity symbol grouping

Input:

$\mathcal{F}$: index of the failure data node;
M: CDM of the erasure code;
m: number of the parity chunks in a strip;
w: number of the symbols in a data/parity chunk

Output:

G: a set of recovery groups for the failure data symbols

```
1: Initialize G = NULL
2: for i = 0 to mw − 1 do
3:     for j = Fw to Fw + w − 1 do
4:         if [M]_{i,j} == 1 then
5:             AddToG(g, d_j, p_i)
6:         end if
7:     end for
8: end for
```

Method

Step 2: recovery solution search

Algorithm 2 Recovery solution search

Input:

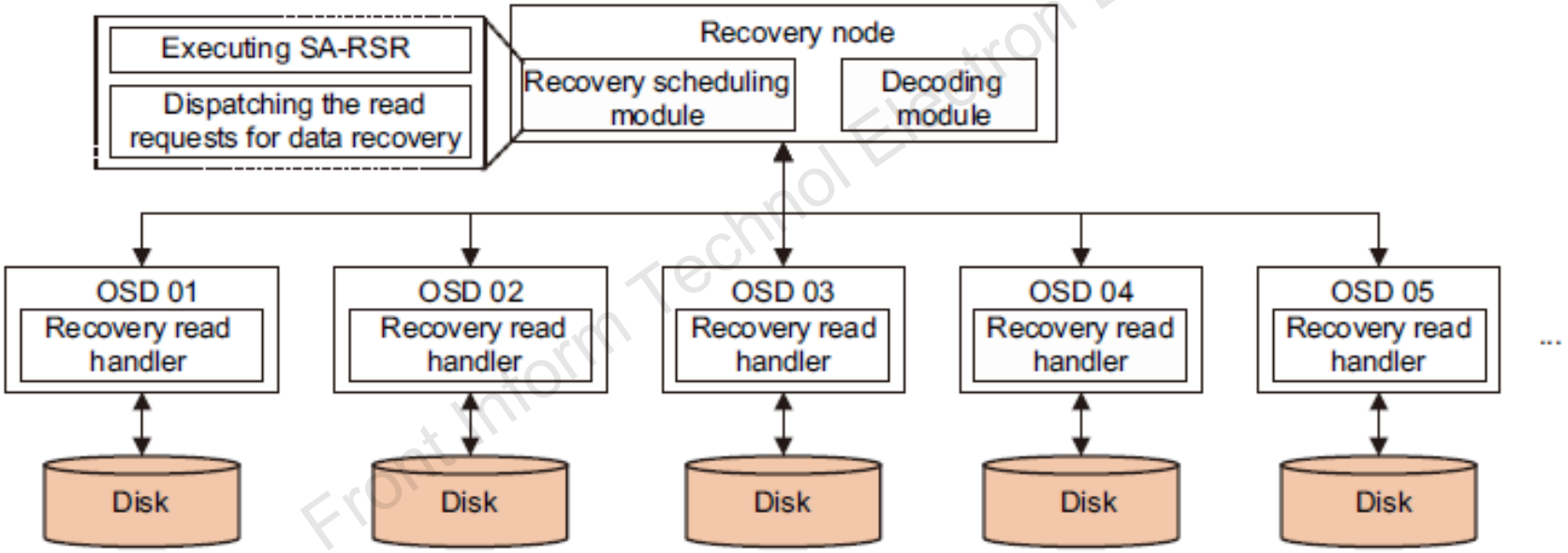
$\mathcal{F}$: index of the failure data node
M: CDM of the erasure code
k: number of data chunks in a stripe
m: number of parity chunks in a stripe
w: number of symbols in a data/parity chunk
G: a set of recovery groups for the failure data symbols generated by Algorithm 1, and g_i is the i^{th} element of G
G_{r}: current recovery solution for the failure data symbols
T_0: initial temperature
T_{end}: termination temperature
K: temperature attenuation coefficient ($K < 1$)
N_{e}: number of external cycle iterations
N_{i}: number of internal cycle iterations

Output:

S: an optimal symbol-reading scheme for recovering failure data symbols

1: Initialize $T = T_0$ and G_{r} =NULL
2: **for** $i = 0$ to $w - 1$ **do**
3: x=one element selected randomly from g_i
4: **while** x is in G_{r} **do**
5: x=one element selected randomly from g_i
6: **end while**
7: G_{r}=G_{r}+x
8: **end for**
9: $S = \text{generate\_solution}(G_{\text{r}})$
10: **for** $i = 0$ to N_{e} **do**
11: **for** $j = 0$ to N_{i} **do**
12: Select an element $p_{\text{tmp\_sid}}$ from mw parity symbols randomly
13: Calculate groups g_{s} to which $p_{\text{tmp\_sid}}$ belongs
14: **if** $p_{\text{tmp\_sid}} \notin G_{\text{r}}$ **then**
15: Select a group $g_{\text{tmp\_gid}} \in g_{\text{s}}$ randomly
16: $G_{\text{r}_{\text{new}}} = \text{replace\_symbol}(G_{\text{r}}, p_{\text{tmp\_sid}}, g_{\text{tmp\_gid}})$
17: $S_{\text{new}} = \text{generate\_solution}(G_{\text{r}_{\text{new}}})$
18: $\Delta = \text{cal\_symbol\_num}(S) - \text{cal\_symbol\_num}(S_{\text{new}})$
19: **if** $\Delta > 0$ or $\exp(\Delta/T) >$ rand()/rand_max **then**
20: $G_{\text{r}} = G_{\text{r}_{\text{new}}}$ and $S = S_{\text{new}}$
21: **end if**
22: **end if**
23: **end for**
24: $T = TK$
25: **if** $T < T_{\text{end}}$ **then**
26: break
27: **end if**
28: **end for**
29: **return** S

Architecture of our method in Ceph



Integration of SA-RSR in Ceph

Major results

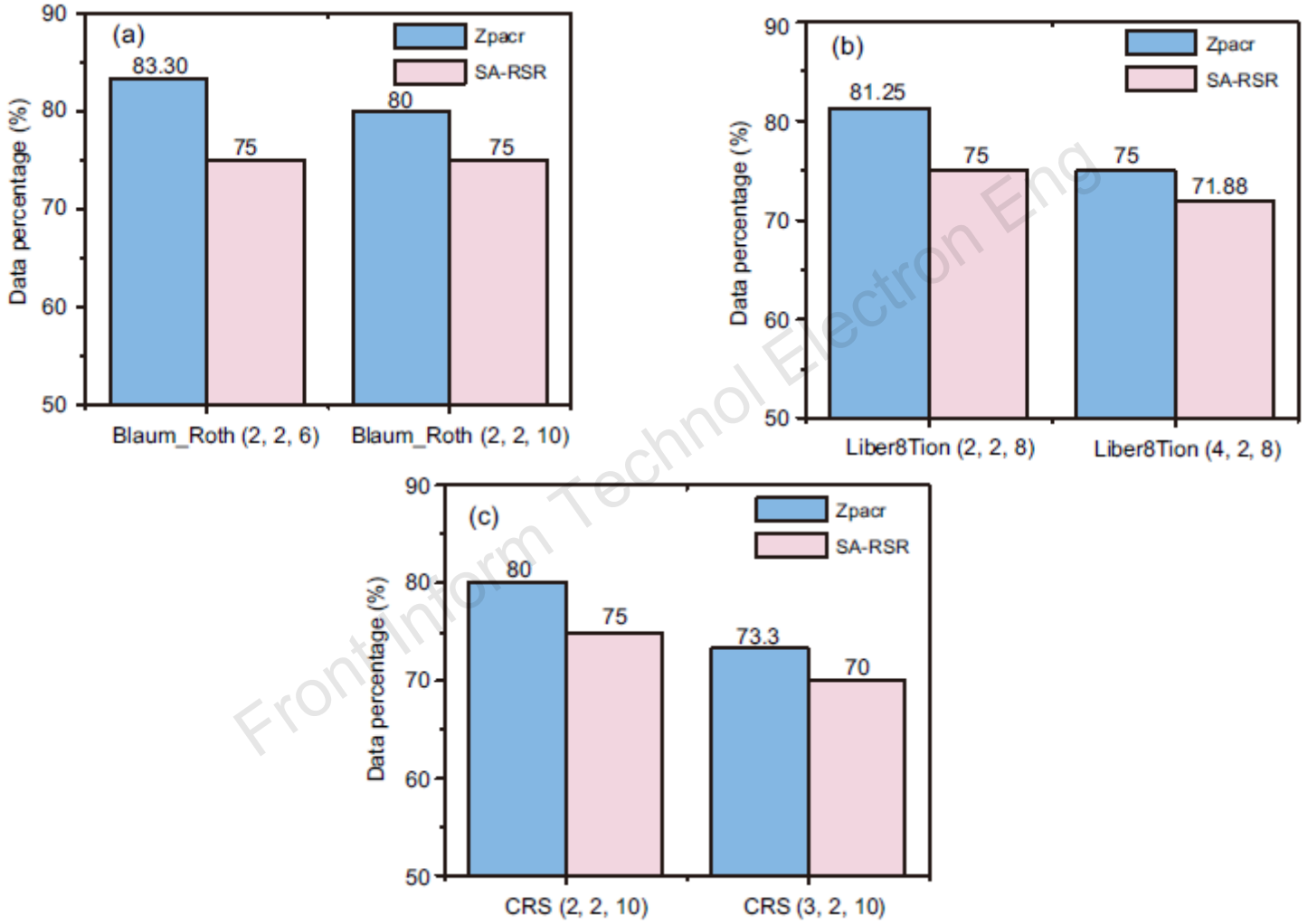


Fig. 7 Percentage of the data needed for Zpacr and SA-RSR over the conventional approach in terms of double-fault-tolerant codes: (a) Blaum_Roth code; (b) Liber8Tion code; (c) CRS code (m = 2)

Major results

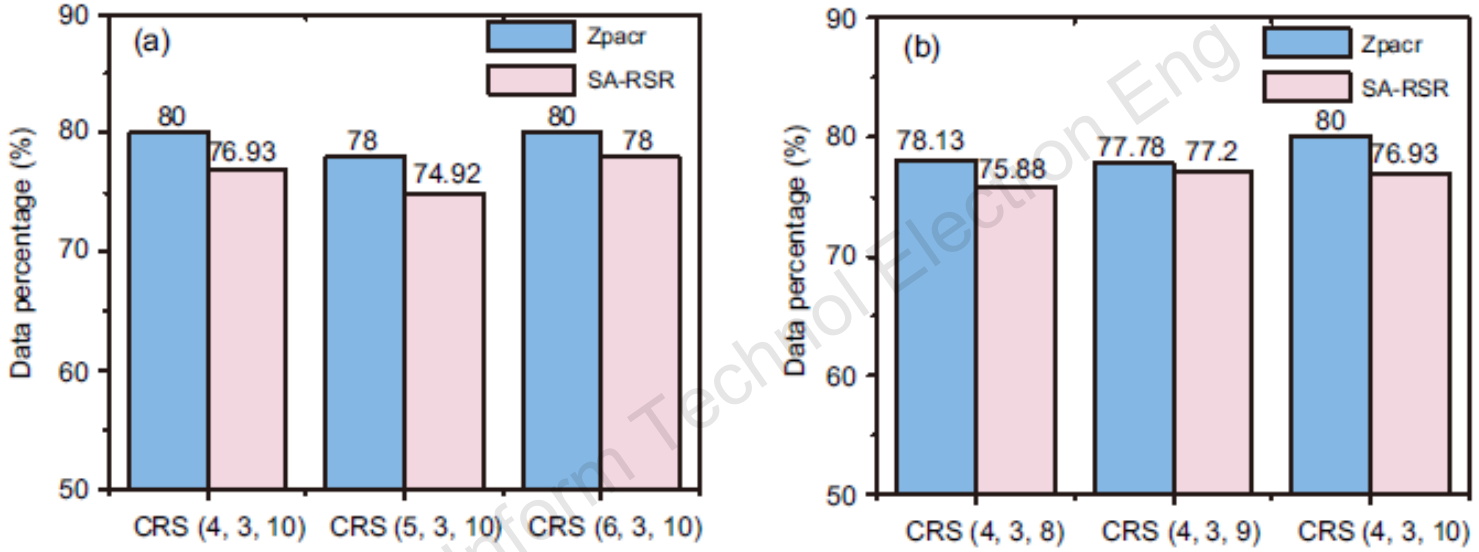


Fig. 8 Percentage of the data needed for Zpacr and SA-RSR over the conventional approach in terms of triple-fault-tolerant codes: (a) CRS code with varying *k* (*m* = 3, *w* = 10); (b) CRS code with varying *w* (*k* = 4, *m* = 3)

Major results

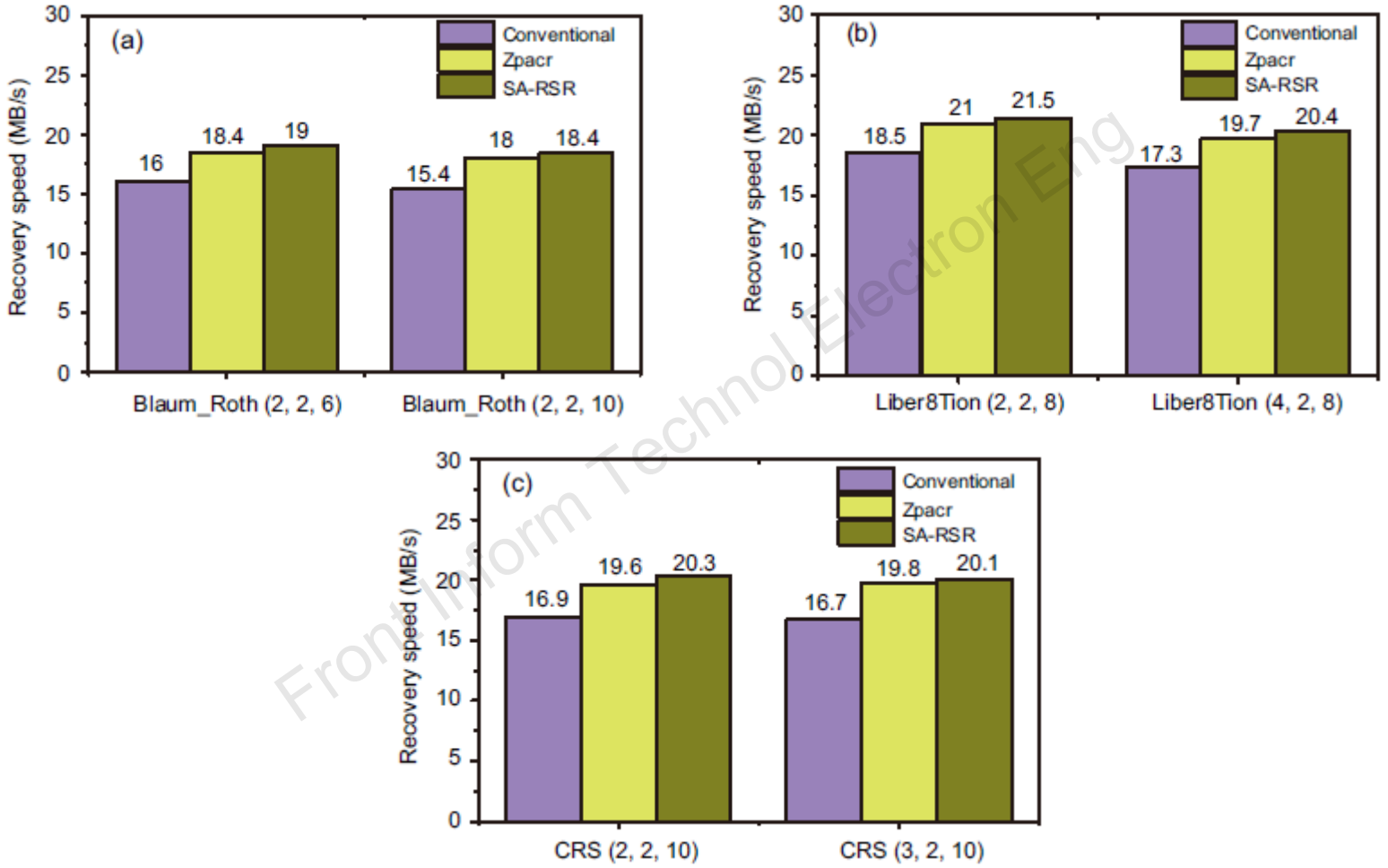


Fig. 11 Recovery speed comparison among the conventional approach, Zpacr, and SA-RSR for different codes that tolerate two node failures: (a) Blaum_Roth code; (b) Liber8Tion code; (c) CRS code (*m* = 2)

Major results

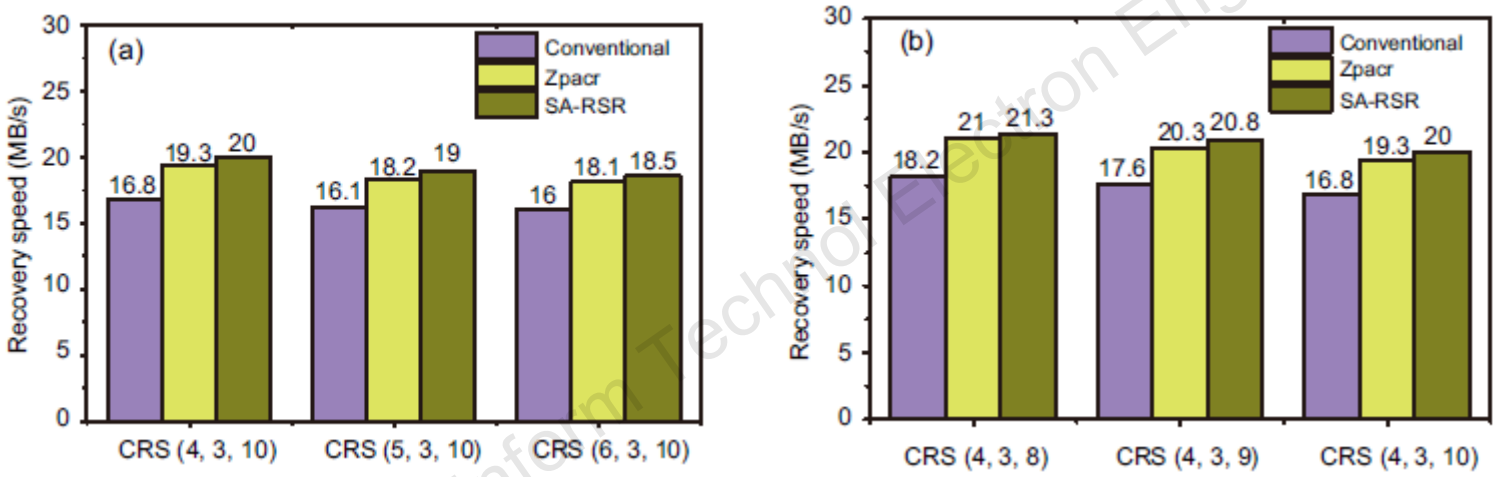


Fig. 12 Recovery speed comparison among the conventional approach, Zpacr, and SA-RSR for different codes that tolerate three or more node failures: (a) CRS code (m = 3, w = 10); (b) CRS code (k = 4, m = 3)

Conclusions

1. We proposed a random search recovery algorithm that can provide optimal recovery performance by minimizing the data needed for single-node failure recovery.

2. We conducted experiments by implementing Zpacr and our algorithm in a real distributed storage system, Ceph.

3. The results showed that our algorithm reduced the amount of data required for single-node failure recovery by 22%–30% and improved the performance of data recovery by 16.22%–20.36% in Ceph when compared to the conventional recovery method.

Xingjun ZHANG received his PhD degree in Computer Architecture from Xi'an Jiaotong University, China, in 2003. From Jan. 2004 to Dec. 2005, he was a post-doctoral fellow at the Computer School of Beihang University, China. From Feb. 2006 to Jan. 2009, he was a research fellow at the Department of Electronic Engineering of Aston University, UK. He is now a full professor and dean of the School of Computer Science & Technology, Xi'an Jiaotong University. His research interests include high-performance computing, big data storage system, and machine learning acceleration.