Wei HU, Guang-ming LIU, Yan-huang JIANG. FTRP: a new fault-tolerance framework using process replication and prefetching for high-performance computing. *Frontiers of Information Technology & Electronic Engineering,* 19(10):1273-1290.
https://doi.org/10.1631/FITEE.1601450

FTRP: a new fault-tolerance framework using process replication and prefetching for high-performance computing

Key words: High-performance computing; Proactive fault tolerance; Failure locality; Process replication; Process prefetching

Corresponding author: Wei HU
E-mail: huwei@nscc-tj.gov.cn

ORCID: Wei HU, http://orcid.org/0000-0002-8839-7748

Motivations

1. Reliability is one of the major challenges as the scale of the supercomputer grows up from petascale to exascale.

2. The checkpoint/restart mechanism is still a reactive mechanism and inherently inefficient.

3. Proactive fault tolerance are often combined with periodic checkpointing. How to effectively choose a countermeasure in both proactive and reactive mechanisms according to the fault tolerance status of an application is a crucial and complicated problem.

Main ideas

1. We present the FTRP framework combining the benefits of proactive and reactive fault tolerance mechanisms to form a whole fault tolerance system.

2. FTRP provides a work-most (WM) cost model to evaluate the system status against failures in real time and adopts the appropriate fault tolerance choice at runtime to minimize the failure impact.

3. Based on the observation and analysis of real parallel systems, we find failure locality in supercomputers.

4. We present a new proactive fault tolerance mechanism called ‘process replication with process prefetching’ (PRP2).

Failure Locality

Definition 1 (Failure temporal locality) The nodes that failed in the immediate past have a high probability of failure in the immediate future.

Definition 2 (Failure spatial locality) The nodes that physically located near a node that failed in the immediate past have a high probability of failure in the immediate future.

FTRP Framework

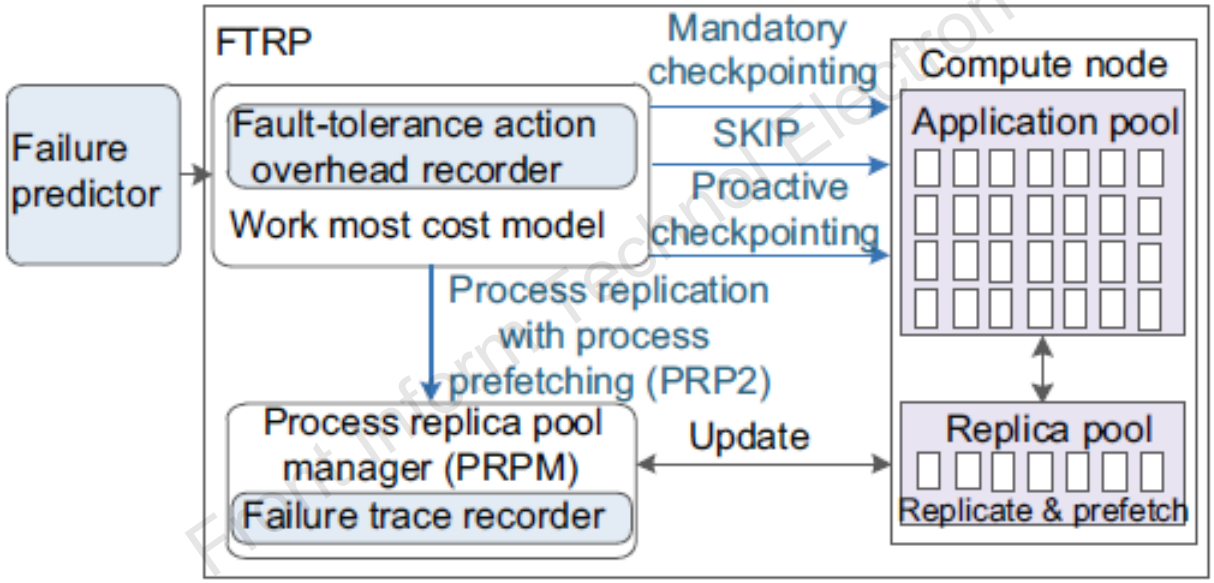

Fig. 4 FTRP framework

Examples of FTRP running process

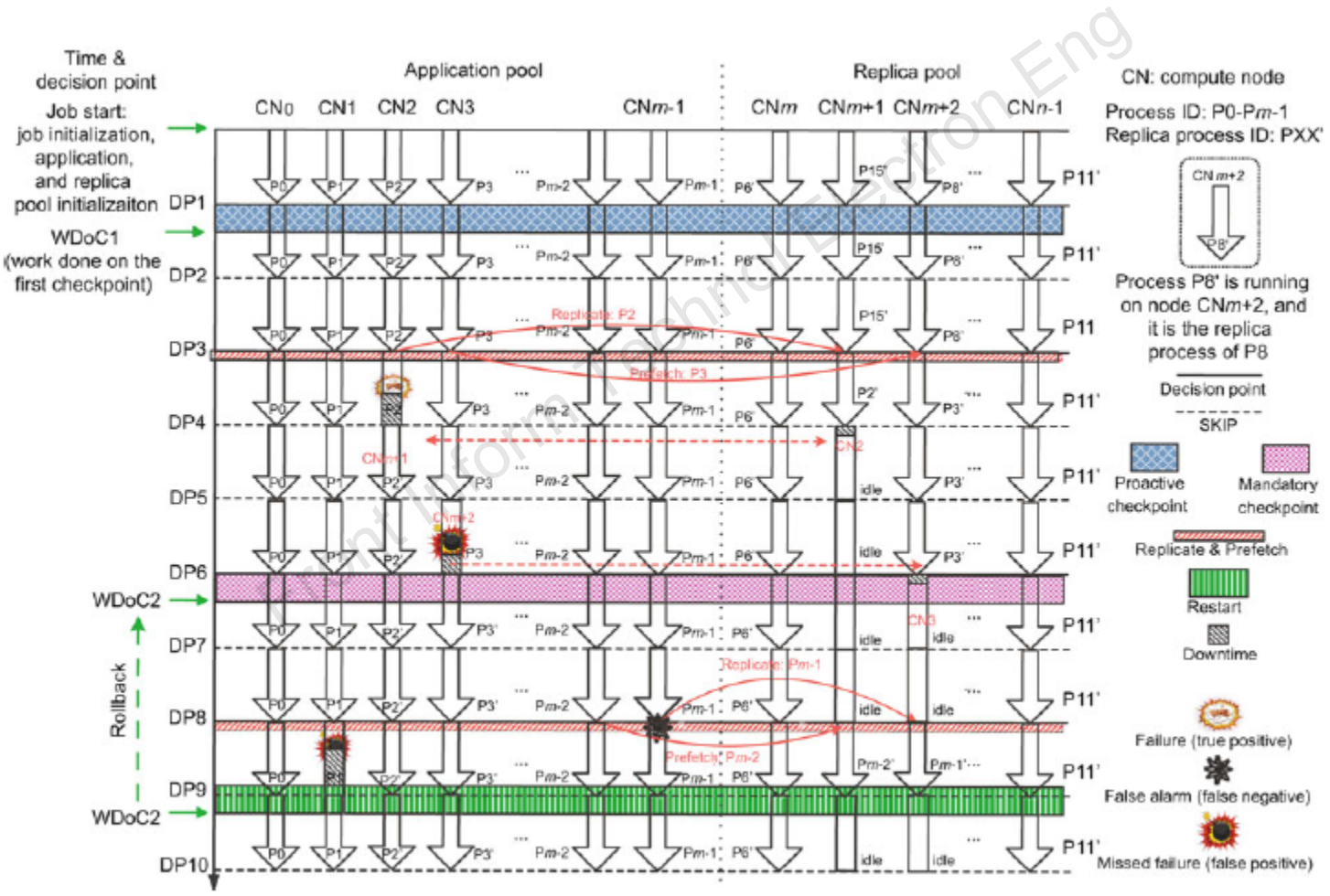

Fig. 5 Examples of the FTRP running process

Major results

1. Impact of prediction accuracy

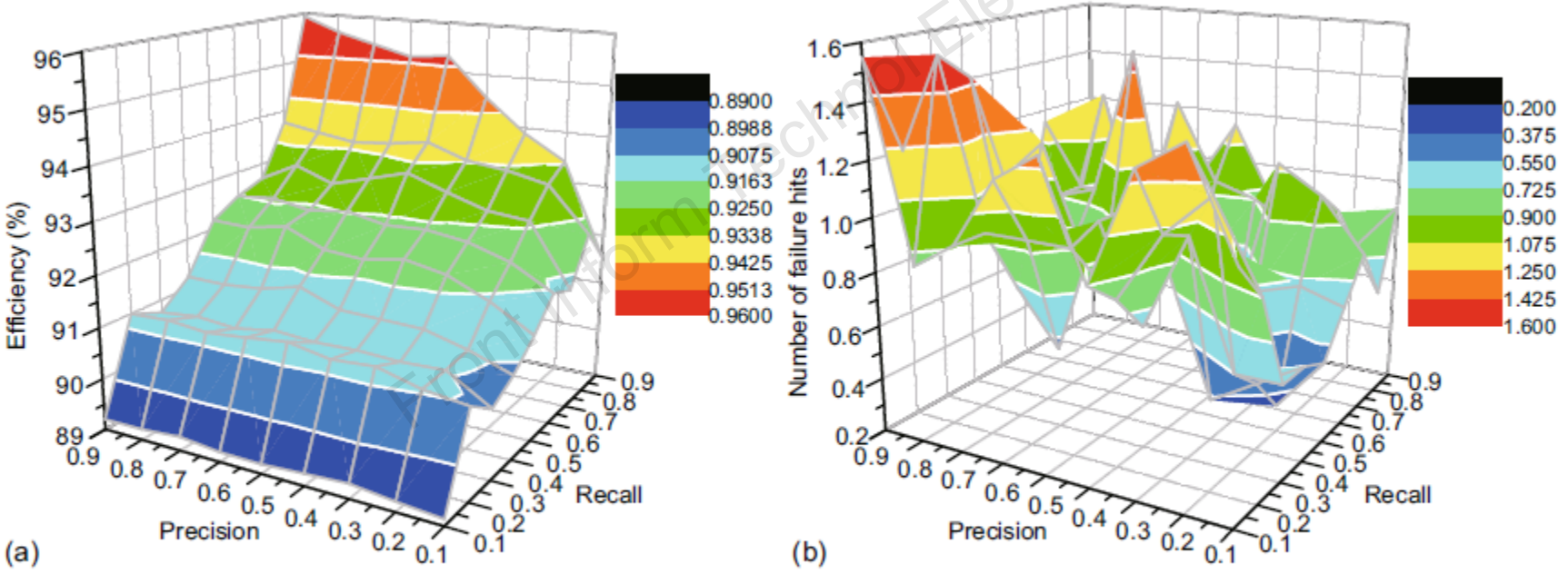

Fig. 8 Impact of prediction accuracy using PNNL08 traces: (a) application efficiency under different precision and recall; (b) corresponding number of failure hits in the replica pool

Major results

1. Impact of prediction accuracy

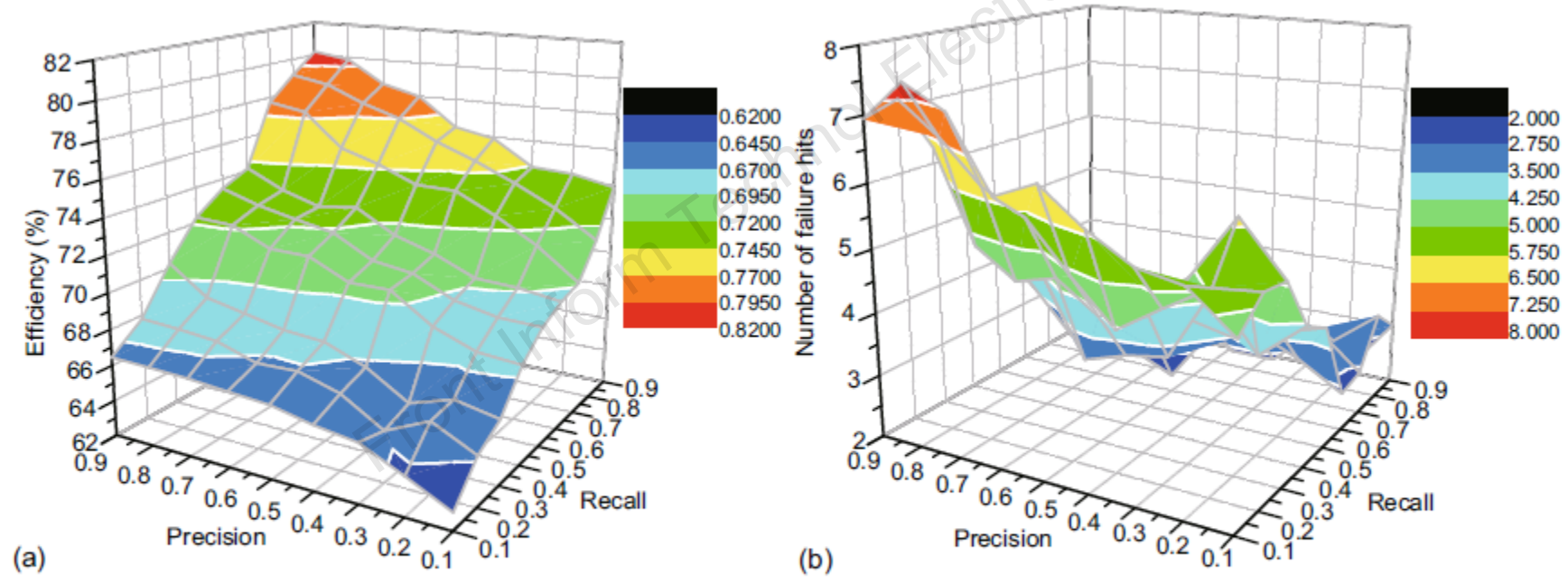

Fig. 9 Impact of prediction accuracy using THNSCC traces: (a) application efficiency under different precision and recall; (b) corresponding number of failure hits in the replica pool

Major results

2. Impact of replication overhead

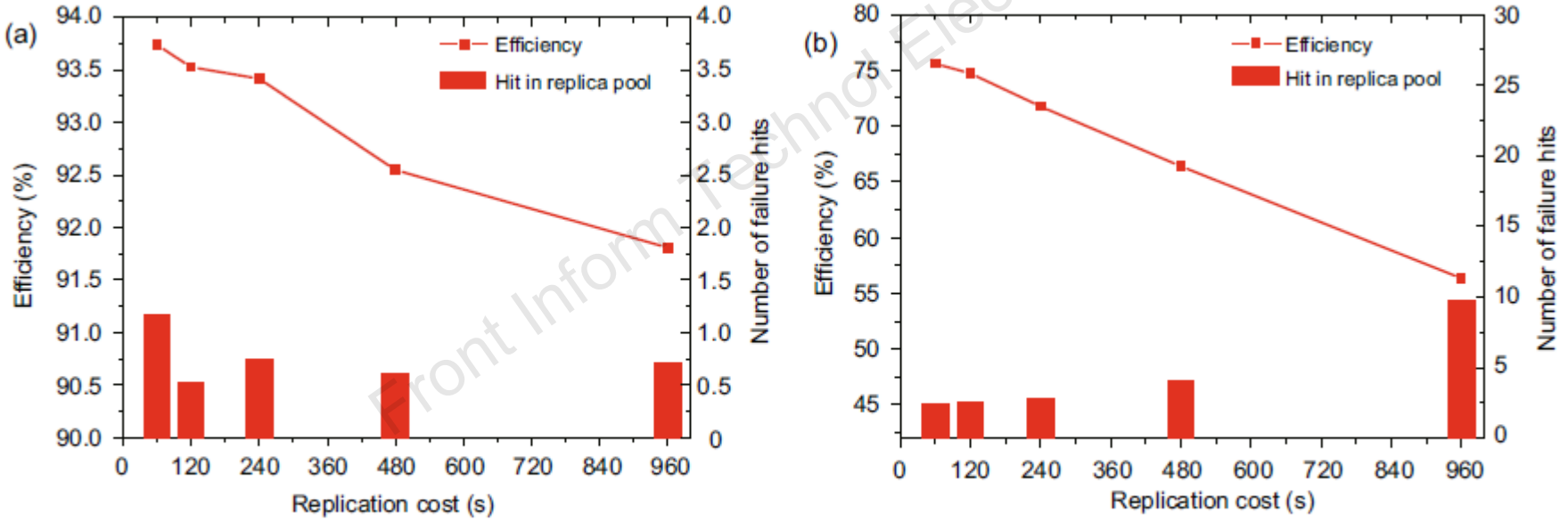

Fig. 10 Impact of replication overhead when the replication overhead ranges from 60 s, 120 s, 240 s, and 480 s to 960 s: (a) PNNL08 traces; (b) THNSCC traces

Major results

3. Impact of stride length

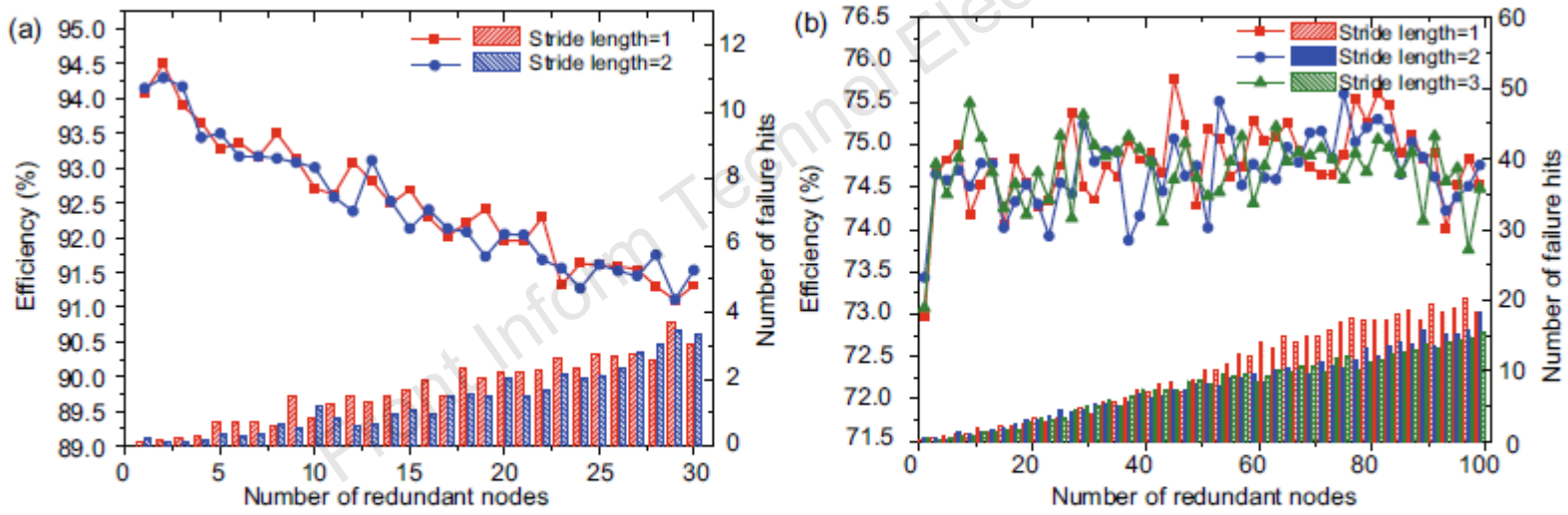

Fig. 11 Impact of stride length

Due to the map sheet limit, compared with (a), (b) illustrates only the results of odd redundant node numbers

Major results

4. Application efficiency comparison

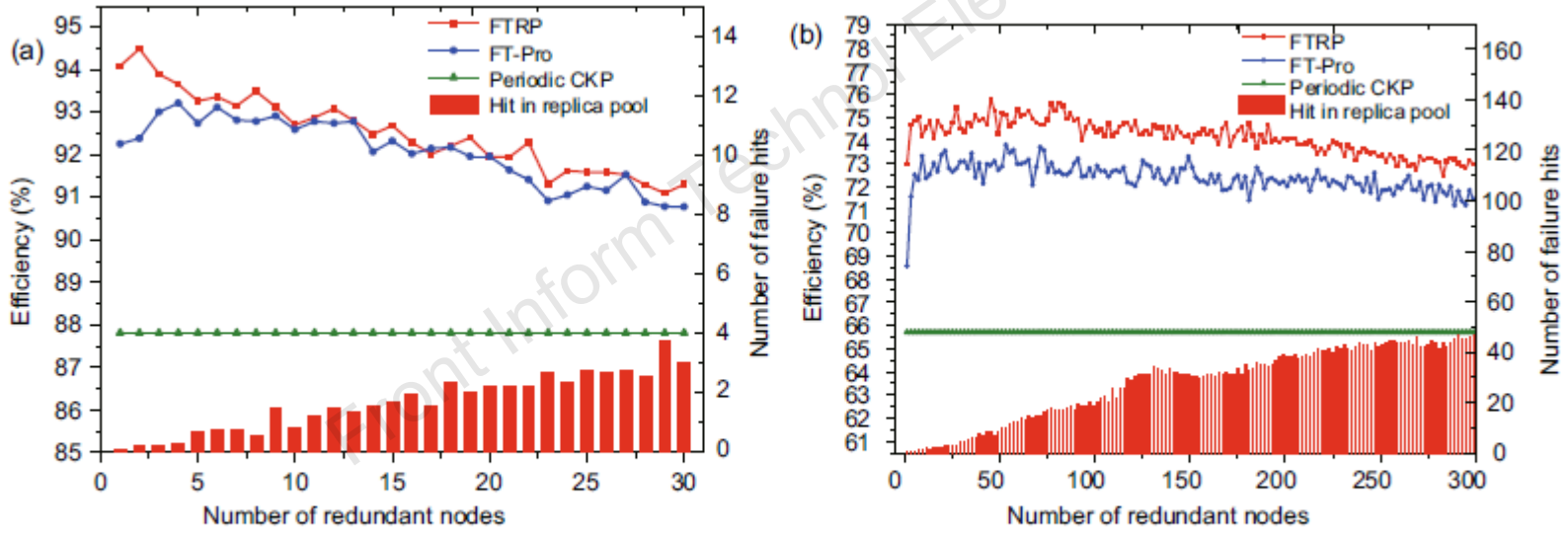

Fig. 12 Application efficiency comparison among three fault tolerance mechanisms including FTRP, FT-Pro, and periodic checkpointing: (a) PNNL08 traces; (b) THNSCC traces

Major results

5. Application efficiency on different computation scales

Table 3 Settings of experiments on different computation scales

N_{nodes}	N_s	C_{chk}(s)	C_r(s)	C_{rep}(s)	Precision	Recall
704	4	240	240			
1408	7	253	253			
2112	11	275	275			
2816	14	308	308	120	0.7	0.7
3520	18	360	360			
4224	21	430	430			
4928	25	510	510			
5632	28	600	600			

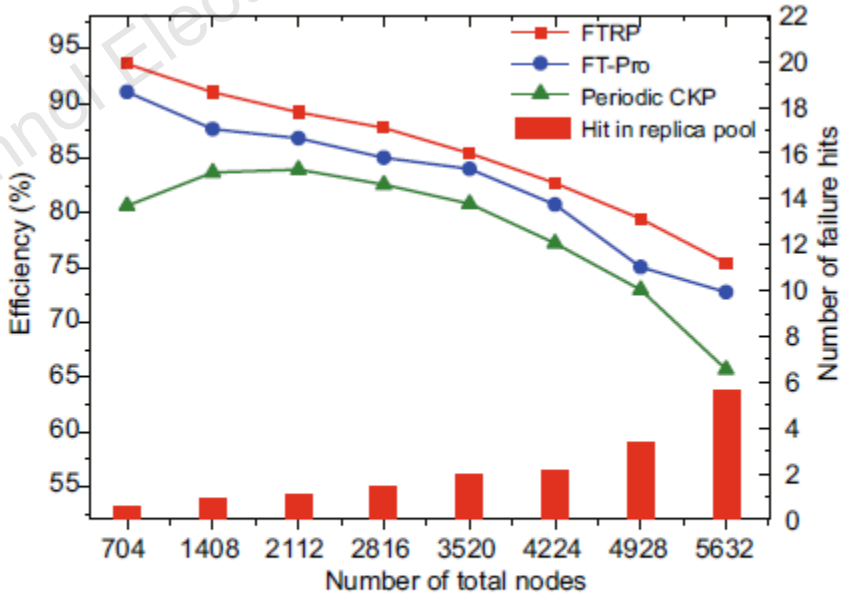

Fig. 13 Application efficiency on different computation scales

Conclusions

1. FTRP requires a less accurate failure prediction with the help of PRP2 and failure locality (FTRP outperforms periodic checkpointing when the precision is more than 0.1 and the recall is more than 0.1 or 0.3 for different systems). The process prefetching method using failure locality effectively improves the flexibility of the framework.

2. An allocation of redundant nodes (0.5%–1%) is efficient for FTRP to reach the above performances.

3. FTRP has the potential to achieve a high performance for larger supercomputer systems.