A Content-Aware Block Placement Algorithm for Reducing PRAM Storage Bit Writes

Brian Wongchaowart Marian K. Iskander Sangyeun Cho

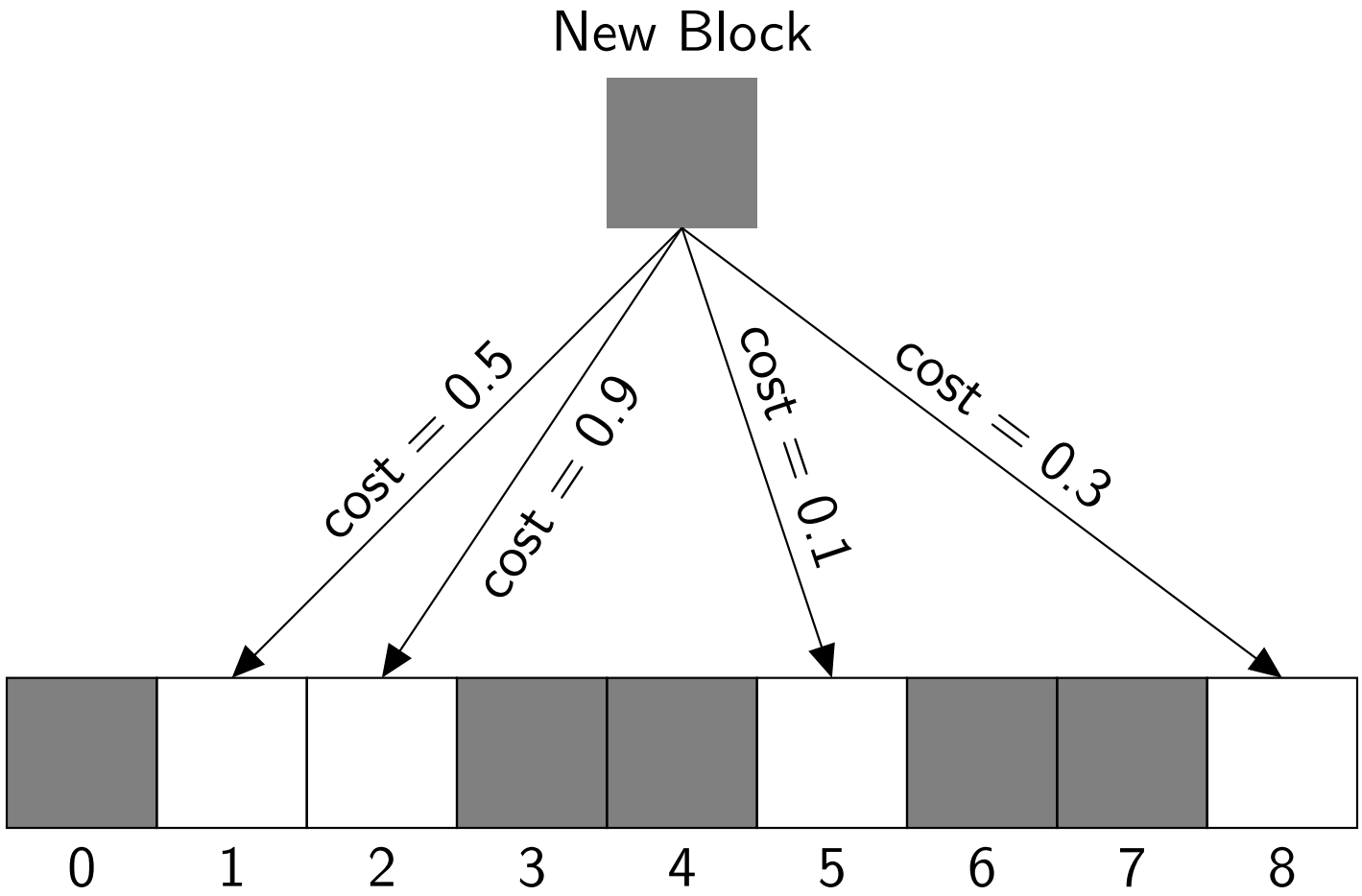

University of Pittsburgh

26th IEEE Symposium on Massive Storage Systems and Technologies
May 6–7, 2010

Motivation

- Consider a block storage device in which the cost of overwriting a block depends on both the existing block contents and the new block contents.
- There are too many free blocks available for overwriting to compare the cost of overwriting each one individually.
- How can one do better than picking an arbitrary free block to overwrite?

Block Placement Decision



PRAM Storage

- Nonvolatile Phase-change Random Access Memory (PRAM) may soon replace flash memory and DRAM in many applications.
- Each memory cell contains a material that has two phases with very different electrical properties.
- An "amorphous phase" exhibits high resistivity, while a "crystalline phase" has much lower resistivity.
- Reading the bit value stored in a cell consists of sensing its resistivity (a fast, low-power operation).

PRAM Updates

- In order to change the bit value stored in a PRAM cell, the phase-change material must be brought into a different phase by heating.
- Heating the phase-change material to its crystallization temperature for a sufficiently long period of time causes it to assume its crystalline state.
- Heating it to a yet higher temperature for a short period of time makes the material amorphous.
- Both of these operations require high-power current pulses (relative to the read operation).

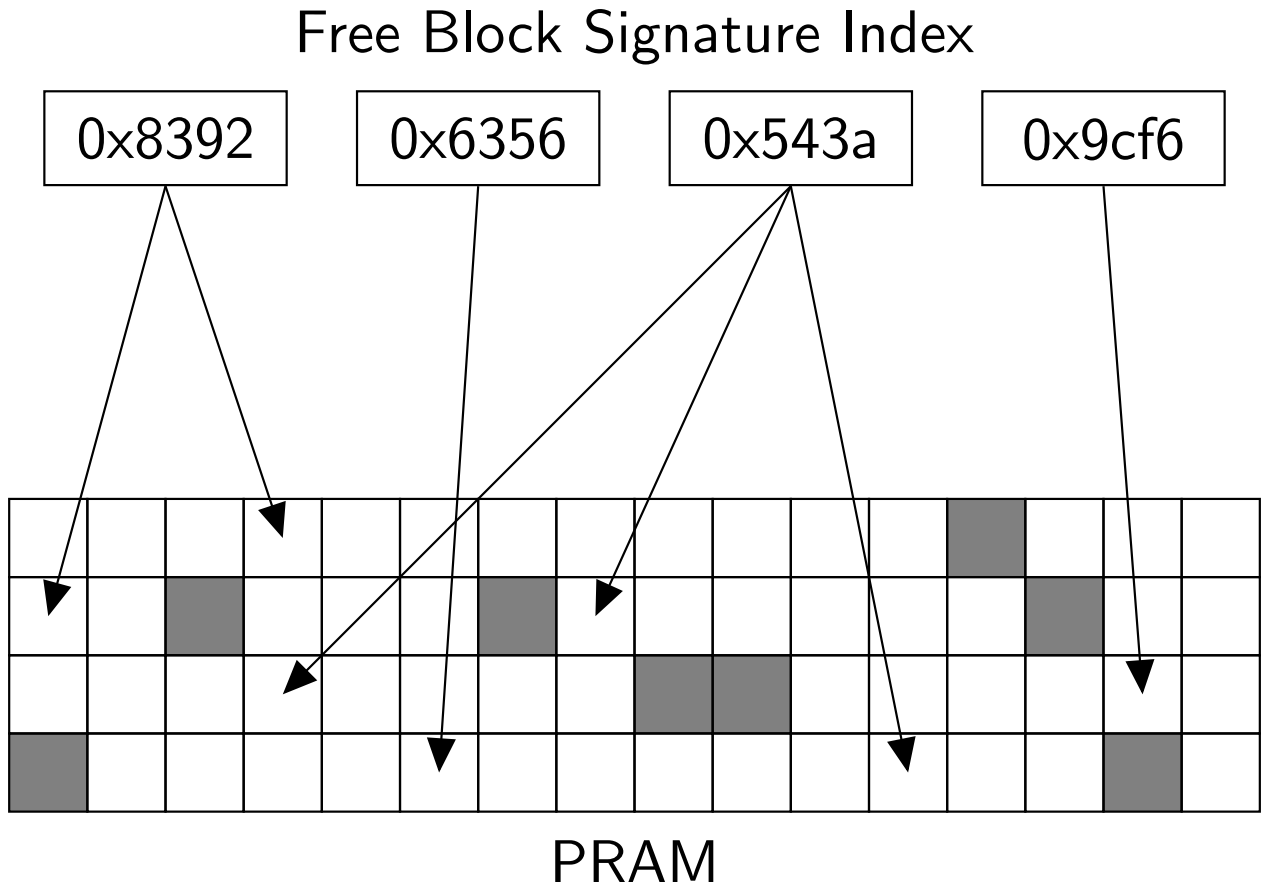
Differential Writes

- Individual PRAM cells can be programmed independently of other cells.
- When overwriting data stored in PRAM, only cells whose current value differs from the new value to be written need to be updated.
- Since programming a cell requires an order-of-magnitude more energy than reading it, it is advantageous to read every cell and only program those that need to be updated. This technique is known as data-comparison write (DCW).
- Key point: writing a block of data to a free location with similar contents (small Hamming distance) is less expensive.

Our Approach: Content-Based Block Signatures

- Our idea is to index all of the free blocks in the PRAM using a content-based signature.
- When a new data block needs to be written, its signature is computed as well.
- Blocks with matching signatures have similar contents, so a free location can be chosen for a new data block by looking up the signature of the new data in the free block index.

Free Block Signature Index



Block Signature Computation

- Divide a block into n equal-size sets of bits and count the number of 1-bits in each set.

- The block signature is a vector containing the approximate number of 1-bits in each set. Element i of this vector is an m-bit integer that denotes the range of bit counts in which the actual number of 1-bits in set i lies.
- Example: each set consists of 128 bits and $m = 4$, so the possible values for the number of 1-bits in a set are divided into 16 ranges. An approximate bit count of 0 stands for 0–7 bits, an approximate bit count of 1 stands for 8–15 bits, etc.

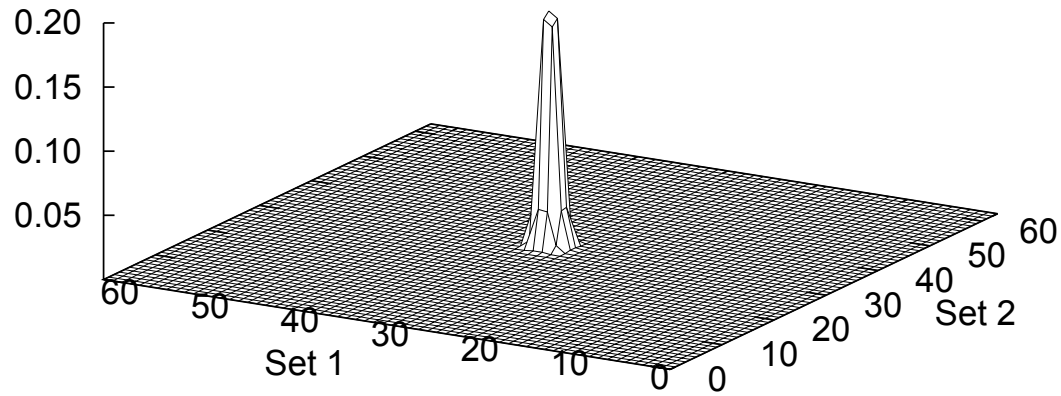


Figure: Distribution of signature values for uniformly distributed random data when there are 2 sets per block and 6 bits are used in the signature to represent the approximate number of 1-bits in each set.

Block Placement Algorithm

- Assume that the signature of every free block has been indexed.
- When a request to write a new data block arrives at the PRAM, the signature of the new block is computed.
- If the PRAM contains at least one free block with the same signature value as the new block, pick the free block that is most similar to the new block from among the first ℓ free blocks with the same signature, where ℓ is a parameter called the *search distance limit*.
- If there are no free blocks with the same signature value as the block to be written, arbitrarily pick a signature value that is present in the index and pick from among the first ℓ free blocks with that signature.

Evaluation

- We use a fixed block size of 512 bytes and block signature sizes of 16 and 32 bits.
- The number of sets per block varies from 1 to the number of bits in the signature.
- We consider search distance limits of 1, 5, and 10. A search distance limit of 1 means that the block placement decision is made using the free block signature index alone without the need to read and compare the contents of multiple free blocks in the PRAM.
- We compare the number of bits written by our signature-based block placement algorithm with the number of bits written using differential writes without block placement optimization (DCW alone) and the number of bits written if the free block most similar to the block to be written is always chosen (exhaustive search).

Random Trace

- The PRAM initially has 128 MB of free space containing uniformly distributed random data.
- 64 MB of uniformly distributed random data is then written to the PRAM. We simulate the block placement decisions and calculate the fraction of the bits in the write requests that actually needs to be written.
- We do not expect placement optimization to be effective for random or compressed data where the data blocks are uniformly distributed over the space of possible data blocks because the PRAM is unlikely to contain a free block that is similar to a new data block to be written.

Sets/block	Bits/set	Total bits	Search distance limit		
			1	5	10
1	16	16	49.99	49.08	48.79
2	8	16	49.97	49.07	48.78
4	4	16	49.97	49.06	48.77
8	2	16	49.94	49.03	48.73
16	1	16	49.88	49.28	49.25
1	32	32	49.99	49.08	48.79
2	16	32	49.98	49.11	48.86
4	8	32	49.96	49.38	49.30
8	4	32	49.93	49.04	48.75
16	2	32	49.88	49.28	49.25
32	1	32	49.89	49.89	49.89

Table: Percentage of the random trace requiring a bit write. DCW alone: 50.00%. Exhaustive search: 46.47%.

Permutation Trace

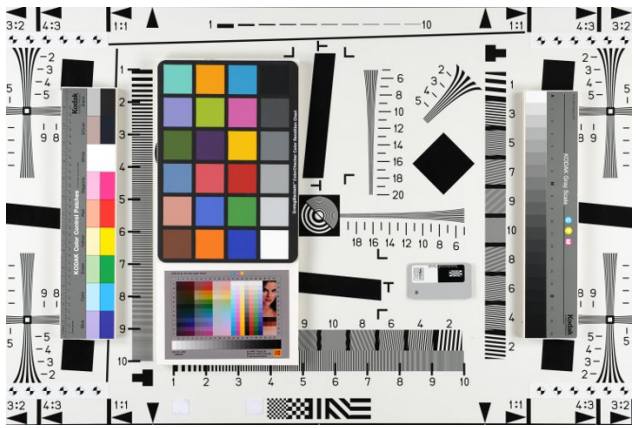
- As in the random trace, the PRAM initially has 128 MB of free space containing uniformly distributed random data.
- A random sample of half of the free blocks in the PRAM is written back to the PRAM in random order.
- Ideally, no bits at all should be written, since all the blocks in the write requests are already contained in the PRAM.

Sets/block	Bits/set	Total bits	Search distance limit		
			1	5	10
1	16	16	49.94	48.84	48.34
2	8	16	49.90	48.71	48.07
4	4	16	49.96	49.04	48.73
8	2	16	49.88	48.80	48.27
16	1	16	37.68	3.81	0.05
1	32	32	49.94	48.84	48.34
2	16	32	46.96	36.86	28.43
4	8	32	34.17	10.57	2.91
8	4	32	49.48	47.19	46.48
16	2	32	37.68	3.81	0.05
32	1	32	0.00	0.00	0.00

Table: Percentage of the permutation trace requiring a bit write. DCW alone: 50.00%. Exhaustive search: 0.00%.

JPEG Trace

- A 256 MB digital camera PRAM memory card initially contains a FAT32 file system populated with 188 MB of JPEG images. Free blocks are zeroed.
- The images on the card are deleted (but not overwritten), and then 170 MB of different JPEG images are written to simulate a user taking new photos.
- We use test shots from a digital SLR camera review.[1]



[1] http://www.imaging-resource.com/PRODS/D3X/D3XTHMB.HTM

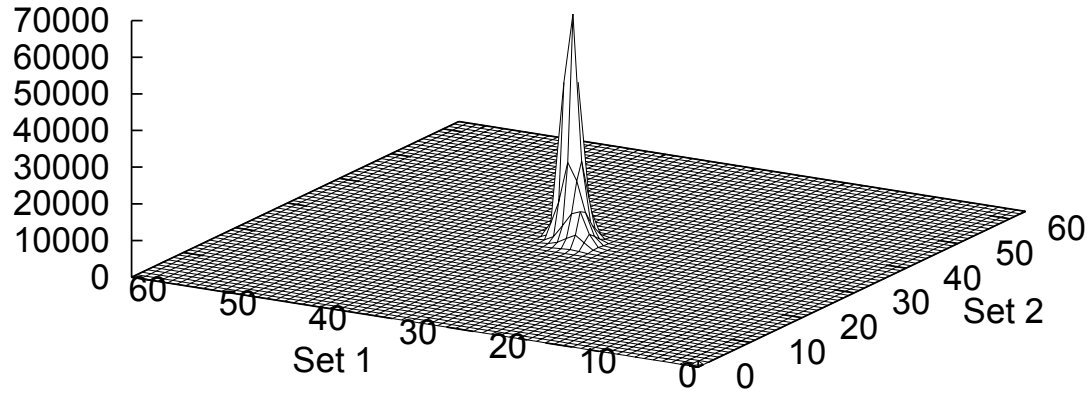


Figure: Distribution of signature values for the write request data blocks of the JPEG trace.

Sets/block	Bits/set	Total bits	Search distance limit		
			1	5	10
1	16	16	49.53	48.64	48.35
2	8	16	49.54	48.64	48.36
4	4	16	49.55	48.65	48.36
8	2	16	49.56	48.66	48.37
16	1	16	49.57	48.83	48.65
1	32	32	49.53	48.64	48.35
2	16	32	49.54	48.74	48.54
4	8	32	49.54	49.06	49.01
8	4	32	49.52	48.64	48.36
16	2	32	49.50	48.77	48.59
32	1	32	49.54	49.52	49.52

Table: Percentage of the JPEG trace requiring a bit write. DCW alone: 49.78%. Exhaustive search: 46.06%.

DNG Trace

- This trace is similar to the JPEG trace, except that uncompressed Adobe Digital Negative (DNG) versions of the same images are used to simulate the raw image formats favored by professional photographers.
- The PRAM represents a 1 GB memory card that initially contains a FAT32 file system populated with 946 MB of DNG images.
- The old images are deleted and 757 MB of new DNG images are written.

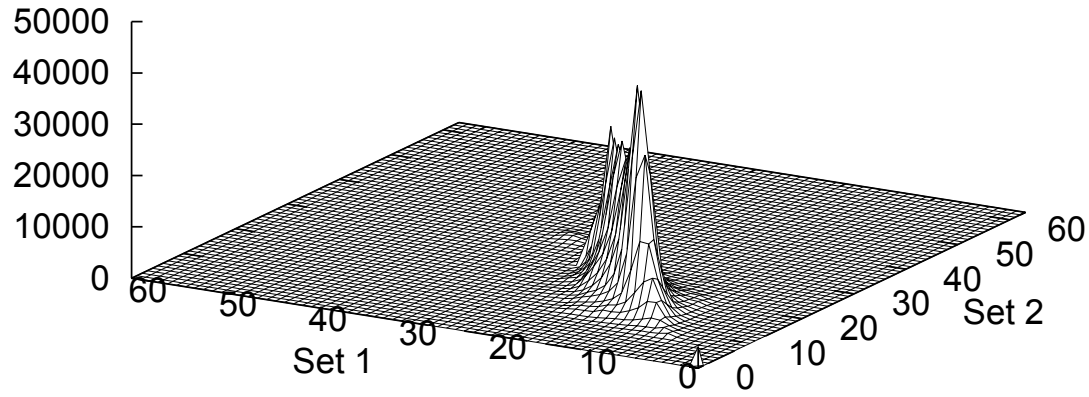


Figure: Distribution of signature values for the write request data blocks of the DNG trace.

Sets/block	Bits/set	Total bits	Search distance limit		
			1	5	10
1	16	16	30.34	29.39	28.99
2	8	16	30.26	29.31	28.92
4	4	16	30.29	29.54	29.31
8	2	16	30.57	30.00	29.78
16	1	16	32.55	31.86	31.36
1	32	32	30.34	29.39	28.99
2	16	32	30.25	29.00	28.66
4	8	32	30.29	29.64	29.57
8	4	32	30.31	29.43	29.10
16	2	32	30.44	29.68	29.33
32	1	32	32.52	31.61	31.29

Table: Percentage of the DNG trace requiring a bit write. DCW alone: 32.51%. Exhaustive search on a 1/8 sample of the trace: 25.02%.

Kernel Build Trace

- Five Linux 2.6.31 kernels are compiled from the source code using different configuration settings.
- All blocks in the file system are initially free and zeroed.
- The files generated by each build are deleted after the build completes, at which point the next build is started.

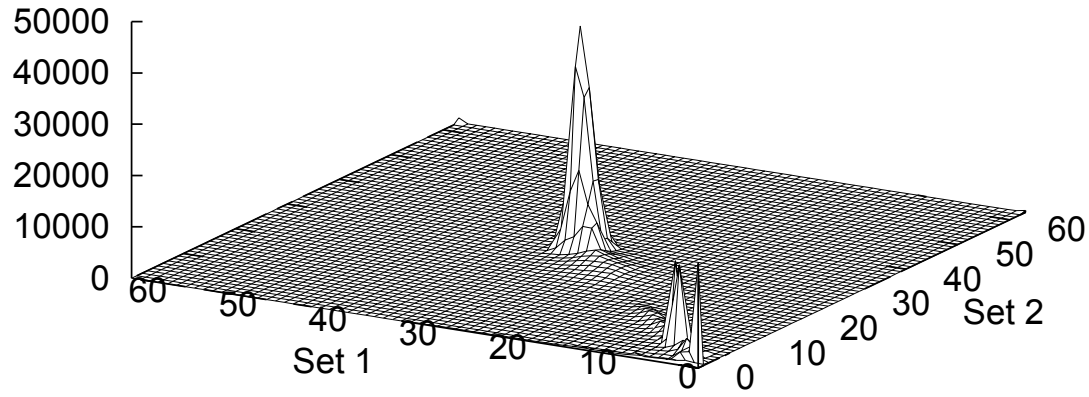


Figure: Distribution of signature values for the write request data blocks of the kernel build trace.

Sets/block	Bits/set	Total bits	Search distance limit		
			1	5	10
1	16	16	33.87	32.63	32.08
2	8	16	33.20	32.19	31.68
4	4	16	33.20	32.39	32.01
8	2	16	33.94	32.83	32.29
16	1	16	32.03	30.68	30.27
1	32	32	33.87	32.63	32.08
2	16	32	32.35	31.29	31.10
4	8	32	31.57	30.97	30.90
8	4	32	32.53	31.51	30.98
16	2	32	31.96	30.60	30.20
32	1	32	30.46	30.13	30.09

Table: Percentage of the kernel build trace requiring a bit write. DCW alone: 38.21%. Exhaustive search: 23.82%.

Suspend-to-Disk Trace

- A Linux system was booted, the user checked his email, and then the system was suspended to a dedicated disk partition using the Linux 2.6.31 swsusp (software suspend) implementation.
- The contents of the suspend partition at this point are used as the initial state of the PRAM.
- The system was resumed, the user checked his email again, and then the system was suspended a second time. The trace of write requests consists of the data written to disk by this second suspend operation.

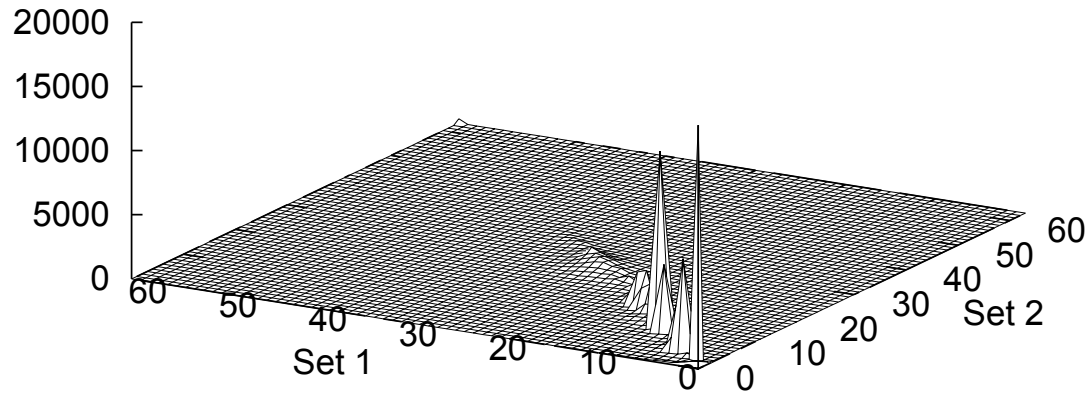


Figure: Distribution of signature values for the write request data blocks of the suspend-to-disk trace.

Sets/block	Bits/set	Total bits	Search distance limit		
			1	5	10
1	16	16	7.75	3.05	2.54
2	8	16	4.46	2.27	2.08
4	4	16	9.44	4.26	3.26
8	2	16	12.52	8.57	6.77
16	1	16	12.26	8.57	8.00
1	32	32	7.75	3.05	2.54
2	16	32	2.27	2.17	2.16
4	8	32	2.01	1.96	1.95
8	4	32	3.11	2.02	1.87
16	2	32	6.52	4.46	3.85
32	1	32	7.08	5.44	4.91

Table: Percentage of the suspend-to-disk trace requiring a bit write. DCW alone: 16.09%. Exhaustive search: 0.32%.

NAS Parallel Benchmark Snapshot Traces

- High-performance computing applications often generate data snapshots to avoid restarting in case of system failure.
- We use data snapshots from four NAS Parallel Benchmark programs: BT (block tridiagonal solver), CG (conjugate gradient), FT (FFT), and MG (multigrid).
- Each benchmark program writes four snapshots to the PRAM storage device. The first and second snapshots are written, the space occupied by the first snapshot is freed and the third snapshot is written, and finally the space occupied by the second snapshot is freed and the fourth snapshot is written.
- There are initially 3 GB of free space in the PRAM and we experiment with five possibilities for the contents of this space: all 0-bits, and as many copies of the snapshots produced by the BT, CG, FT, and MG programs, respectively, as are required to fill up the free space.

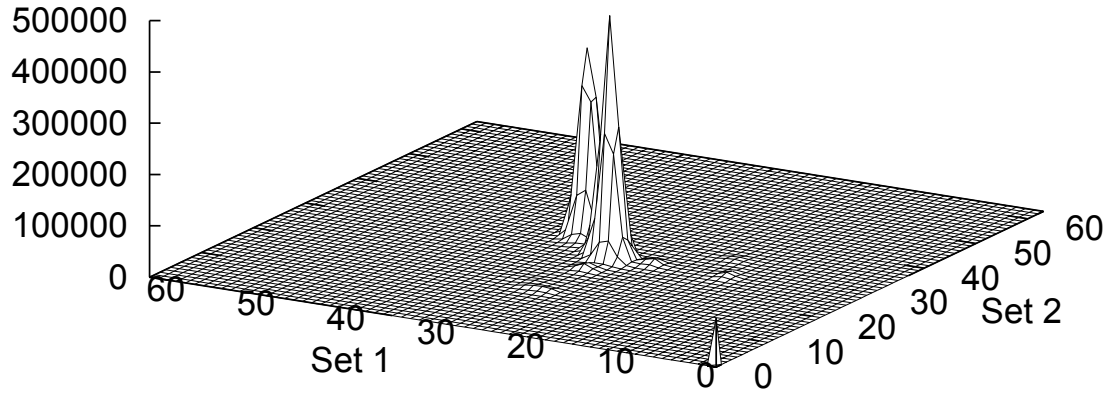


Figure: Distribution of signature values for the write request data blocks of the BT trace.

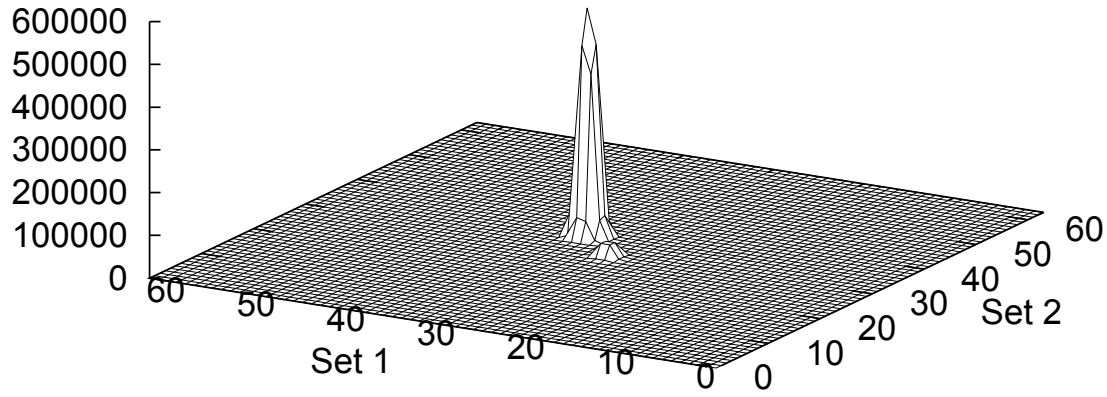


Figure: Distribution of signature values for the write request data blocks of the CG trace.

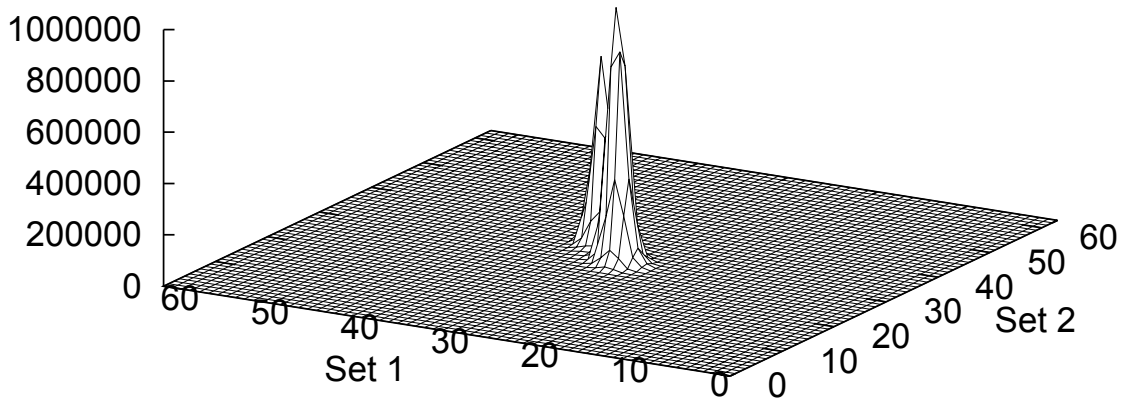


Figure: Distribution of signature values for the write request data blocks of the FT trace.

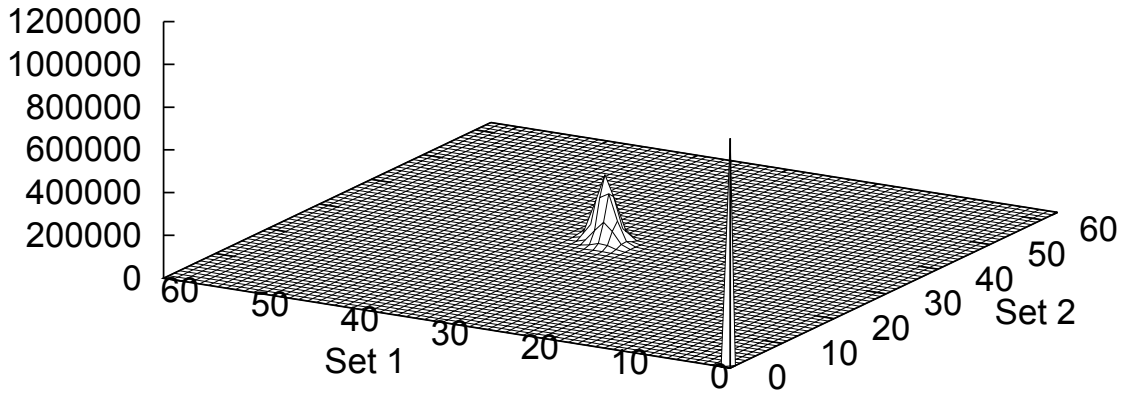


Figure: Distribution of signature values for the write request data blocks of the MG trace.

Trace	Initial PRAM contents				
	Zeros	BT	CG	FT	MG
BT	47.81	49.65	50.37	49.67	49.31
CG	51.16	49.13	43.46	49.47	49.57
FT	48.93	49.38	50.00	48.81	49.35
MG	36.29	48.38	49.81	48.84	43.93

Table: Percentage of the NAS snapshot traces requiring a bit write when DCW is used with random block placement.

Trace	Initial PRAM contents				
	Zeros	BT	CG	FT	MG
BT	42.24	0.00	43.89	43.27	42.48
CG	32.96	39.45	0.00	34.38	40.14
FT	42.84	41.34	43.56	16.06	42.67
MG	33.65	40.29	41.85	41.55	0.00

Table: Percentage of the NAS snapshot traces requiring a bit write when signature-based block placement is used with 4 sets per block, 8 bits per set, and a search distance limit of 1.

Trace	Initial PRAM contents				
	Zeros	BT	CG	FT	MG
BT	38.27	14.70	39.91	38.83	39.15
CG	26.67	25.66	0.38	25.14	25.81
FT	42.27	42.05	43.46	17.90	42.78
MG	33.15	39.64	40.42	40.03	15.04

Table: Percentage of the NAS snapshot traces requiring a bit write when DCW is used with a manual block placement strategy: overwrite the first snapshot with the third snapshot and the second snapshot with the fourth snapshot.

Directions for Future Work

- Quantitative analysis of energy savings and latency impact using measured values for the set, reset, and read operations.
- Evaluation using traces that represent mixed workloads.
- Applications beyond mass storage, e.g., PRAM main memory behind a DRAM cache.
- Operating system hints, e.g., write an updated block to its current location (skip placement optimization).

Summary

- The number of bit programming operations needed to store a new data block in a PRAM storage device depends on the current contents of the location at which the block is written.
- We proposed a signature-based block placement algorithm for reducing the number of bit writes, which saves energy. Parallelization can save time, but only reducing the number of bit writes saves energy.
- With the right parameter settings, our block placement algorithm was able to reduce the number of bit writes needed to as low as 12.5% of the number needed when DCW (differential writes) alone is used. This figure was achieved without reading and comparing multiple free blocks.

Questions/Comments