علوم رایانش و فناوری اطلاعات
نشریه علمی انجمن کامپیوتر ایران
مجلد ۱۷، شماره ۱، ۱۳۹۸
صفحات ۹۳-۸۴
مقاله عادی

افزایش طول عمر حافظه نهان ورودی/خروجی با استفاده از معماری ترکیبی چندسطحی

مصطفی هادی‌زاده[۱]، رضا سالخورده حقیقی[۲]، حسین اسدی[۳*]

*نویسنده مسئول، دریافت: ۹۸/۰۳/۲۲، بازنگری: ۹۸/۰۸/۱۹، پذیرش: ۹۸/۰۹/۰۲

[۱] دانش‌آموخته کارشناسی ارشد، مهندسی کامپیوتر، دانشکده‌ی مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران
[۲] دانش‌آموخته دکتری، مهندسی کامپیوتر، دانشکده مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران
[۳] استاد، مهندسی کامپیوتر، دانشکده‌ی مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران

چکیده

استفاده از دیسک حالت جامد به‌عنوان حافظه نهان در زیرسامانه‌های ذخیره‌سازی داده یکی از رویکردهای رایج جهت افزایش کارایی این سامانه‌ها است. دوام محدود دیسک‌های حالت جامد، یکی از اصلی‌ترین چالش‌های موجود در طراحی حافظه‌های نهان ورودی/خروجی کارا به شمار می‌آید. در این مقاله، ابتدا تأثیر معماری‌های یک‌سطحی و چندسطحی بررسی و نشان داده می‌شود استفاده از دیسک‌های حالت جامد به‌عنوان حافظه نهان ورودی/خروجی یک‌سطحی، تعداد دفعات نوشتن بر روی این ادوات را نسبت به درخواست‌های نوشتن ارسالی از لایه بالاتر به‌صورت میانگین ۳٫۷ برابر می‌کند که باعث کاهش شدید طول عمر دیسک حالت جامد می‌شود. مبتنی بر این مشاهده، یک معماری ترکیبی شامل DRAM با دوام بالا به‌عنوان سطح اول و دیسک حالت جامد با قابلیت ذخیره‌سازی داده به‌صورت غیر فرار به‌عنوان سطح دوم پیشنهاد می‌شود. در معماری پیشنهادی، فقدان‌های خواندن در سطح اول حافظه نهان درج می‌شوند و به‌منظور بهبود کارایی حافظه نهان چندسطحی در مقایسه با معماری تک‌سطحی، یک فیلتر تنزل پیشنهاد می‌شود که وظیفه مهاجرت صفحات پرطرفدار را از سطح اول به سطح دوم حافظه نهان بر عهده خواهد داشت. ارزیابی‌های انجام شده نشان می‌دهد که معماری پیشنهادی با سربار قابل‌چشم‌پوشی کارایی (به‌طور میانگین ۰٫۳٪) می‌تواند تا ۹۳٪ تعداد دفعات نوشتن بر روی دیسک حالت جامد را کاهش دهد.

کلمات کلیدی: زیرسامانه ذخیره‌سازی، دیسک حالت جامد، حافظه نهان ترکیبی چندسطحی.

۱- مقدمه

در سال‌های اخیر، تولید داده رشد قابل‌ملاحظه‌ای داشته است. میزان داده تولیدشده تا سال ۲۰۱۳ برابر با ۴٫۴ زتابایت بوده است و پیش‌بینی می‌شود با روند فعلی تولید داده، میزان داده تولیدشده تا سال ۲۰۲۰ به رقم ۴۴ زتابایت برسد [۱]. این روند رو به رشد تولید داده، چالش‌های مختلفی مانند قابلیت اطمینان[۱]، زمان دسترسی به داده‌ها و هزینه را در حوزه ذخیره‌سازی داده به وجود آورده است. ازاین‌رو اهمیت زیرسامانه‌های ذخیره‌سازی داده بیش‌ازپیش شده است.

کارایی[۲] پایین ادوات ذخیره‌سازی داده نسبت به سایر بخش‌های سامانه‌های رایانه‌ای، یکی از چالش‌های اصلی سامانه‌های رایانه‌ای در سال‌های اخیر بوده است. رشد سریع کارایی پردازنده و حافظه اصلی و رشد کندتر دیسک‌های سخت بـه سبب وجود بخش‌های مکانیکی موجب شده است دیسک‌های سخت به گلوگاه کارایی سامانه‌ها تبدیل شده و بر کارایی کل سامانه تأثیر بگذارند [۲۱]. ازاین‌رو ارائه روش-هایی برای بهبود کارایی زیرسامانه‌های ذخیره‌سازی داده ازجمله مباحث پرطرفدار در حوزه علوم و مهندسی رایانه به‌حساب می‌آید.

دیسک‌های حالت جامد[۳] (SSD)، ادوات[۴] ذخیره‌سازی داده مبتنی بر حافظه‌های فلش هستند که داده را به‌صورت دائمی ذخیره می‌کنند. گسترش و پیشرفت دیسک‌های حالت جامد، آن‌ها را به گزینه‌هایی جذاب برای استفاده در زیرسامانه‌های ذخیره‌سـازی داده تبدیل کرده است. برخلاف دیسک‌های سخت، دیسک‌های حالت جامد با خصوصیاتی نظیر نداشتن بخش‌های مکانیکی، مصرف توان کم‌تر و زمان پاسخ‌دهی کوتاه‌تر [۲] می‌توانند باعث بهبود کلی عملکرد سامانه شوند. علی‌رغم این خصوصیات مثبت، عواملی مانند قیمت بالا و طول عمر

پایین‌تر دیسک‌های حالت جامد (به سبب دوام[5] محدود حافظه‌های فلش) مانع از جایگزینی کامل آن‌ها با دیسک سخت شده است. ازاین‌رو استفاده از دیسک حالت جامد به‌عنوان بخشی از زیرسامانه ذخیره‌سازی داده مدنظر طراحان سامانه‌های رایانه‌ای و پژوهشگران قرار گرفته است.

استفاده از دیسک حالت جامد به‌عنوان حافظه نهان در زیرسامانه‌های ذخیره‌سازی داده یکی از رویکردهای مطرح‌شده در پژوهش‌های پیشین است. این رویکرد به طراحان زیرسامانه‌های ذخیره‌سازی اجازه می‌دهد تا با صرف هزینه کم‌تر، از کارایی بهتر دیسک حالت جامد بهره ببرند [۴۷]. ازجمله اصلی‌ترین چالش‌های موجود در حوزه استفاده از دیسک حالت جامد به‌عنوان حافظه نهان دیسک سخت، یافتن یک روش مدیریتی بهینه آگاه از دوام به‌منظور افزایش طول عمر دیسک حالت جامد است [۳][۴].

در این مقاله، ابتدا تأثیر سیاست‌های مدیریتی بر روی طول عمر حافظه‌های نهان زیرسامانه ذخیره‌سازی مورد ارزیابی قرار می‌گیرد. مطابق ارزیابی‌های انجام‌گرفته، نوشتن فقدان‌های خواندن بر روی دیسک حالت جامد، تعداد دفعات نوشتن بر روی دیسک حالت جامد را نسبت به درخواست‌های نوشتن لایه بالاتر، به‌طور میانگین ۳,۷ برابر می‌کند. این در حالی است که فقدان‌های خواندن از لحاظ قابلیت اطمینان، بحرانی نیستند و نیازی نیست که در حافظه غیرفرار[6] درج شوند. سپس بر مبنای این نتایج و به‌منظور کاهش اثر فقدان‌های خواندن، یک معماری ترکیبی شامل یک حافظه با دوام بالا (مانند DRAM) به‌عنوان حافظه نهان سطح اول و دیسک حالت جامد به‌عنوان حافظه نهان سطح دوم ارائه می‌شود. در معماری پیشنهادی، فقدان‌های خواندن، که از لحاظ قابلیت اطمینان بحرانی نیستند، تنها بر روی سطح اول حافظه نهان نوشته می‌شود و دیسک حالت جامد داده‌های دست‌خورده[7] را ذخیره می‌کند. همچنین در جهت افزایش کارایی از یک فیلتر تنزل[8] (DF) (DF) به‌منظور درج فقدان‌های خواندن بر روی دیسک حالت جامد استفاده می‌شود.

به‌منظور ارزیابی، حافظه DRAM برای سطح اول حافظه نهان و دیسک حالت جامد به‌عنوان سطح دوم حافظه نهان در نظر گرفته‌شده و یک شبیه‌ساز مبتنی بر ردگیری[9] پیاده‌سازی می‌شود. بررسی‌های انجام شده بر روی ۱۶ بارکاری نشان می‌دهد که استفاده از معماری ترکیبی، با هزینه قابل‌چشم‌پوشی در حوزه کارایی، می‌تواند به‌طور میانگین ۴۶,۵٪ تعداد دفعات نوشتن بر روی دیسک حالت جامد را کاهش دهد.

به‌طور خلاصه، نوآوری‌ها و مشاهدات اصلی این مقاله به این شرح است:

- در این مقاله یک معماری حافظه نهان ترکیبی دوسطحی ورودی/خروجی[10] (HyTIA) ارائه می‌شود. در این معماری، با استفاده از یک حافظه با دوام بالا به‌عنوان سطح اول حافظه نهان، طول عمر حافظه نهان افزایش می‌یابد. درج فقدان‌های خواندن (داده‌های دست‌نخورده[11]) در سطح اول حافظه نهان می‌تواند تعداد دفعات نوشتن بر روی دیسک حالت جامد (سطح دوم حافظه نهان) را به‌طور چشم‌گیری (تا ۹۳٪) کاهش دهد.
- در معماری پیشنهادی یک فیلتر تنزل ارائه می‌شود تا با بررسی فقدان‌های خواندن، فرآیند مهاجرت فقدان‌های خواندن پرطرفدار به سطح دوم حافظه نهان را مدیریت کند. به‌این‌ترتیب، روش پیشنهادی با کاهش ناچیز نرخ برخورد و بدون کاهش قابلیت اطمینان، تعداد دفعات نوشتن بر روی دیسک حالت جامد را به‌طور میانگین ۴۶,۵٪ کاهش می‌دهد.
- با بررسی جامع ۱۶ بارکاری مطرح در حوزه ذخیره‌سازی داده، مشاهده می‌شود عدم مدیریت مناسب حافظه نهان در زیرسامانه‌های ذخیره‌سازی داده می‌تواند بیش از درخواست‌های نوشتن‌های ارسال‌شده توسط لایه بالاتر بر روی طول عمر حافظه نهان اثر بگذارد. سیاست‌های مدیریتی که با هدف افزایش کارایی اقدام به درج فقدان‌های خواندن در دیسک حالت جامد می‌کنند، به‌طور میانگین، تعداد دفعات نوشتن بر روی دیسک حالت جامد نسبت به تعداد درخواست‌های نوشتن را ۳,۷ برابر می‌کند.

در ادامه این مقاله، بخش ۲ به پیش‌زمینه‌های موردنیاز می‌پردازد. در بخش ۳ پژوهش‌های پیشین مورد ارزیابی قرار می‌گیرند. در بخش ۴ روش پیشنهادی بیان خواهد شد. در بخش ۵ راهکار ارائه‌شده مورد ارزیابی قرار خواهد گرفت و سرانجام در بخش ۶ به نتیجه‌گیری و کارهای آتی خواهیم پرداخت.

۲- پیش‌زمینه

دیسک حالت جامد با ساختاری غیر مکانیکی، کارایی بالاتری نسبت به دیسک سخت ارائه می‌دهد. به‌منظور نشان دادن تفاوت بین دیسک‌های سخت و دیسک‌های حالت جامد، ابتدا به بررسی کارایی و قیمت ادوات مختلف در حوزه ذخیره‌سازی داده خواهیم پرداخت. سپس عواملی مانند ساختار داخلی دیسک‌های حالت جامد، سطوح دسترسی و فرایندهای مدیریتی این ادوات بررسی می‌شوند. همچنین رویکرد استفاده از دیسک حالت جامد به‌عنوان حافظه نهان دیسک سخت نیز در این بخش بررسی خواهد شد.

۲-۱- بررسی کارایی و هزینه ادوات ذخیره‌سازی داده

برخلاف دیسک‌های سخت، دیسک‌های حالت جامد مبتنی بر حافظه فلش به سبب نداشتن تأخیرهایی مانند زمان یافتن[12] و چرخش[13]، کارایی بالاتری ارائه می‌دهند. شکل ۱ مقایسه‌ای میان کارایی و هزینه ادوات ذخیره‌سازی داده را نشان می‌دهد. دیسک‌های حالت جامد با استفاده از حافظه‌های غیرفرار فلش می‌توانند کارایی در مرتبه ۱۰۰ میکروثانیه (us) ارائه بدهند [۳۲]. این در حالی است که دیسک‌های سخت زمان پاسخی در مرتبه میلی‌ثانیه ارائه می‌دهند [۴۸]. استفاده از دیسک‌های حالت جامد که دارای واسط از نوع PCI Express هستند می‌تواند منجر به بهبود بیشتر کارایی بشود. بااین‌حال استفاده از دیسک‌های حالت جامد مبتنی بر واسط PCI Express نیاز به توسعه بیشتر دستورات سازگار با این فناوری دارد [۳۳].

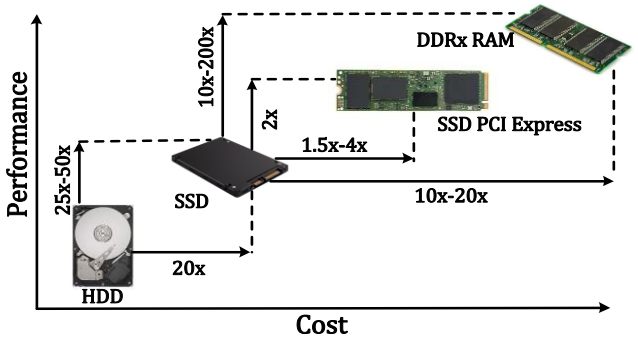
با وجود کارایی بالاتر دیسک‌های حالت جامد، به دلیل دوام پایین حافظه‌های فلش و قیمت پایین‌تر دیسک‌های سخت، این ادوات کماکان گزینه‌هایی محبوب برای ذخیره‌سازی داده به‌حساب می‌آیند و دیسک‌های حالت جامد نتوانسته‌اند کاملاً جانشین آن‌ها شوند. استفاده از فناوری‌های مختلف حوزه ذخیره‌سازی در کنار یکدیگر، این امکان را فراهم می‌کند تا با صرف هزینه کم‌تر، بتوانیم از کارایی بهتر این ادوات استفاده کنیم.

۲-۲- ساختار دیسک‌های حالت جامد

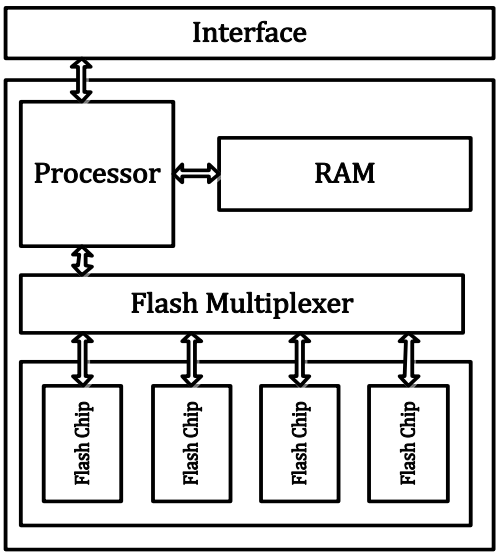
دیسک‌های حالت جامد از حافظه‌های غیرفرار فلش ساخته شده‌اند و برخلاف دیسک سخت فاقد اجزای مکانیکی هستند. این ماهیت غیر مکانیکی سبب شده است دیسک‌های حالت جامد برخلاف دیسک‌های سخت، تأخیرهایی مانند زمان یافتن و چرخش نداشته باشند. به‌منظور افزایش طول عمر و کارایی این ادوات، پژوهش‌های متعددی به بهبود ساختار داخلی دیسک‌های حالت جامد پرداخته‌اند [۳۸][۳۹][۴۰][۴۱]. هر بسته فلش می‌تواند دارای چند تراشه باشد که به آن بستر[14] نیز گفته می‌شود. هر بستر از تعدادی سطح[15] تشکیل شده است. درون هر سطح، تعدادی ثبات و قطعه[16] وجود دارد و هر قطعه خود شامل چندین صفحه[17] است [۵].

شکل ۲ ساختار داخلی یک دیسک حالت جامد را نشان می‌دهد [۵]. در ساختار هر دیسک حالت جامد علاوه بر تراشه‌های فلش، بخش‌هایی مانند پردازنده و حافظه

میانگیر نیز وجود دارد. ازجمله مشکلات حافظه‌های فلش در هنگام نوشتن در یک سلول، نیاز به پاک کردن آن سلول است؛ درواقع برای بازنویسی یک سلول باید حتماً عملیات پاک کردن صورت پذیرد. در دیسک حالت جامد، دسترسی‌های خواندن و نوشتن در سطح صفحه و پاک کردن در سطح قطعه است. در هر عملیات پاک کردن، تمامی صفحات معتبر یک قطعه به قطعه‌ای دیگر کپی شده و سپس کل صفحات آن قطعه پاک می‌شوند که این عملیات، جریمه کارایی زیادی دارد.



شکل ۱- مقایسه بین کارایی و هزینه ادوات ذخیره‌سازی داده [۲۴][۲۵][۳۱]



شکل ۲- ساختار داخلی دیسک حالت جامد [۵]

در کنار جریمه زیاد عملیات پاک کردن، دوام محدود حافظه‌های فلش نیز از دیگر مشکلات دیسک‌های حالت‌جامد است. برای غلبه بر این مشکل، لایه ترجمه فلش (FTL) به‌منظور کاهش تعداد عملیات پاک کردن، در یک درخواست نوشتن ابتدا خانه قبلی را نامعتبر می‌کند و داده جدید را در خانه جدیدی می‌نویسد. در صورت زیاد شدن تعداد خانه‌های نامعتبر، عملیات زباله‌روبی[۱۸] انجام می‌پذیرد تا خانه‌های در دسترس بیشتری در اختیار لایه ترجمه فلش قرار بگیرد. در هنگام انتخاب خانه جدید برای نوشتن، از روش هم‌سطحی فرسودگی[۱۹] استفاده می‌شود. در هم‌سطحی فرسودگی، نوشتن داده‌ها بین خانه‌های مختلف حافظه پخش می‌شود تا در نتیجه آن، دیسک حالت جامد به‌صورت یکنواخت فرسوده شود [۶].

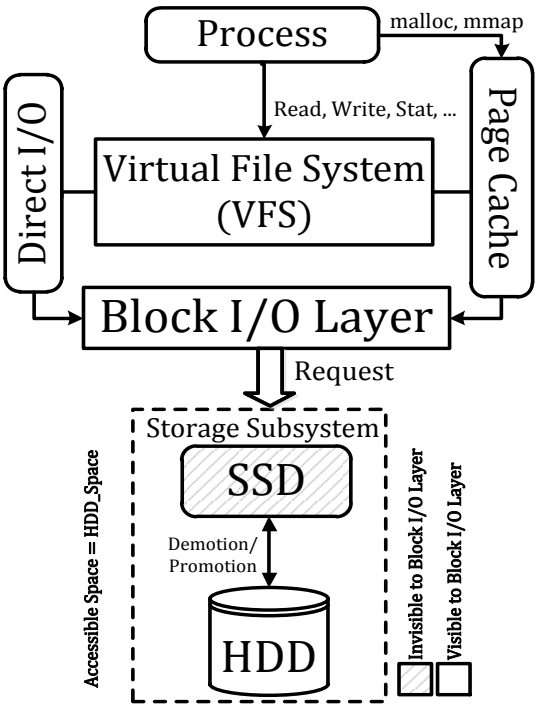
۳-۲- ساختار حافظه نهان ورودی/خروجی

استفاده از دیسک‌های حالت جامد به‌عنوان بخشی از زیرسامانه ذخیره‌سازی این امکان را فراهم می‌کند که با صرف هزینه کم، از مزایای این ادوات استفاده کنیم. پژوهش‌های پیشین، رویکردهای مختلفی به‌منظور استفاده از دیسک حالت جامد در زیرسامانه ذخیره‌سازی پیشنهاد داده‌اند. ازجمله این رویکردها می‌توان به رده‌بندی[۲۰] [۲۲][۳۴][۳۵]، سامانه‌های مجتمع حافظه-دیسک[۲۱] [۲۳][۳۶][۳۷] و استفاده از دیسک حالت جامد به‌عنوان حافظه نهان دیسک سخت [۷][۸][۹] اشاره کرد.

شکل ۳ ساختار حافظه نهان ورودی/خروجی[۲۲] (I/O) را نشان می‌دهد. حافظه نهان به‌منظور جبران فاصله بین کارایی حافظه‌های مختلف در سامانه‌های رایانه‌ای مورد استفاده قرار می‌گیرد و فضای آدرس آن از دید کاربر مخفی است. مطابق شکل ۳ فضای آدرس دستگاه[۲۳] هاشور خورده از دید لایه بالاتر پنهان است. به دلیل کارایی بالاتر دیسک‌های حالت جامد نسبت به دیسک‌های سخت، استفاده از این دیسک‌ها به‌عنوان حافظه نهان در پژوهش‌های پیشین مورد بررسی قرار گرفته است. در پژوهش‌هایی با رویکرد استفاده از دیسک حالت جامد به‌عنوان حافظه نهان، معمولاً یکی از دو سیاست نهان‌سازی درون‌نویسی[۲۴] و پس‌نویسی[۲۵] مورد استفاده قرار گرفته شده است. در حافظه‌های نهان مبتنی بر سیاست پس‌نویسی، با هدف جلوگیری از دسترسی‌های مکرر به دیسک سخت، تمامی درخواست‌های نوشتن در حافظه نهان نوشته شده و پاسخ درخواست ارسال می‌شود.

۳- پژوهش‌های پیشین

ازجمله الگوریتم‌های نهان‌سازی که خصوصیات دیسک حالت جامد را در نظر می‌گیرند، می‌توان به LARC [۸] اشاره کرد. در این روش، یک الگوریتم تنبل[۲۶] بر مبنای الگوریتم ARC پیشنهاد شده است که علاوه بر افزایش دوام دیسک حالت جامد، کارایی را نیز به‌وسیله قرار دادن داده‌های پر ارجاع‌تر در حافظه نهان بهبود می‌دهد. LARC دارای دو صف Q و Q_r می‌باشد که هر دو صف از سیاست LRU[۲۷] پیروی می‌کنند. صف Q متعلق به بلوک‌های حاضر در دیسک حالت جامد است و صف Q_r شامل داده‌هایی با یک‌بار دسترسی است که هنوز روی دیسک حالت جامد قرار نگرفته‌اند. اندازه Q_r به‌صورت پویا تعیین می‌شود. شرط انتقال یک داده به صف Q، برخورد آن داده در صف Q_r است. به‌این‌ترتیب از حضور داده‌هایی با ارجاع کم در حافظه نهان و درنتیجه از آلودگی آن جلوگیری می‌شود.



شکل ۳- ساختار حافظه نهان ورودی/خروجی [۷][۲۶]

ازجمله ضعف‌های الگوریتم LARC می‌توان سرعت واکنش پایین این الگوریتم نسبت به تغییرات مجموعه کاری[۲۸] اشاره کرد. در الگوریتم mARC [۹] به‌منظور رفع این مشکل، روشی مبتنی بر حالت[۲۹] بارکاری پیشنهاد شده که ایده اصلی این

روش، استفاده از الگوریتم‌های نهان‌سازی[۳۰] مختلف برای حالت‌های مختلف است. برای این منظور، مجموعه‌های کاری در قالب یک ماشین حالت مدل‌سازی شده‌اند که این ماشین حالت دارای سه حالت پایدار، ناپایدار و دسترسی یکتا است. در حالت پایدار، نرخ برخورد یک بازه مشخص از دسترسی‌ها تقریباً برابر با بازه قبلی است. زمانی که کاهش نرخ برخورد از یک مقدار آستانه بیشتر شود و نشانه‌هایی از تغییرات در بارکاری بروز پیدا کند، ماشین حالت وارد حالت ناپایدار می‌شود. در حالت دسترسی یکتا، میزان دسترسی‌هایی که برای اولین بار انجام شده زیاد است. الگوریتم mARC در حالت‌های پایدار و دسترسی یکتا از LARC و در حالت ناپایدار از الگوریتم ARC استفاده می‌کند.

در روش uCache [۱۱]، یک حافظه نهان دو سطحی متشکل از DRAM و دیسک حالت جامد با تمرکز بر دو هدف کاهش تعداد نوشتن بر دیسک حالت جامد و کم کردن درخواست‌های نوشتن با اندازه کوچک ارائه شده است. برای هدف اول از یک فیلتر برای تشخیص داده‌های با احتمال دسترسی بیشتر ارائه‌شده (فیلتر uCache) و برای هدف دوم یک حافظه میانگیر (حافظه میانگیر uCache) پیشنهاد شده است. در این روش در صورت رخ دادن یک فقدان حافظه نهان، داده از دیسک سخت خوانده شده و در DRAM نوشته می‌شود. اگر این داده در DRAM برخورد داشته باشد، به هنگام خروج از DRAM در SSD نوشته می‌شود. این روش به دلیل ذخیره کردن داده‌های دست‌خورده در DRAM، قابلیت اطمینان پایینی دارد.

در [۱۲]، چارچوب Me-CLOCK برای پیاده‌سازی حافظه‌های نهان بزرگ با سربار حافظه پایین ارائه شده است. بدین منظور الگوریتم CLOCK به نحوی اصلاح شده است تا اکثر اطلاعات مربوط به حافظه نهان در خود حافظه نهان نگه‌داری و صرفاً ابتدا و انتهای صف FIFO در حافظه اصلی نگه‌داری شود. همچنین به‌جای پرچم‌های استفاده دوباره که در الگوریتم CLOCK استفاده می‌گردد، از یک فیلتر بلوم[۳۱] که ازنظر حافظه سربار کمی دارد، برای ردیابی[۳۲] حضور یا عدم حضور یک بلوک داده در حافظه نهان استفاده شده که این فیلتر بلوم نیز در حافظه اصلی نگه‌داری می‌شود.

در [۱۳]، یک روش نهان‌سازی کارا-نوشتنی[۳۳] مبتنی بر کشش[۳۴] پیشنهاد شده که هدف اصلی آن، قرار دادن داده‌هایی با نرخ برخورد بالا در داخل دیسک حالت جامد و نگه‌داری این داده‌ها در حافظه نهان است. داده زمانی از دیسک حالت جامد خارج می‌شود که محبوبیت آن از یک مقدار آستانه کم‌تر شود. اندازه این آستانه از اندازه دیسک حالت جامد بزرگ‌تر است و داده فرصت بیشتری برای حضور در دیسک حالت جامد خواهد داشت. زمانی که داده بیش از آستانه معین شده در دیسک حالت جامد ماند و برخوردی نداشت، به‌صورت پیش‌فعال[۳۵] از دیسک حالت جامد خارج می‌شود.

در روش SLA-Cache [۱۴]، یک حافظه نهان انتخابی[۳۶] و آگاه از چینش[۳۷] برای فایل سیستم‌های موازی پیشنهاد شده است که از تعداد کمی دیسک حالت جامد به‌عنوان حافظه نهان فایل سرور مبتنی بر دیسک سخت استفاده می‌کند. روش پیشنهادی با هدفِ قرار دادن داده‌های بحرانی ازنظر کارایی بر روی دیسک حالت جامد، مدلی برای محاسبه هزینه دسترسی به داده‌ها ارائه داده است. همچنین سه چینش مختلف برای استفاده از دیسک حالت جامد در نظر گرفته شده تا داده‌ها بر اساس میزان بحرانیتِ کارایی و چینشی که بهترین کارایی را ارائه می‌دهد، بر روی دیسک حالت جامد قرار بگیرند.

معماری ReCA [۷]، یک معماری مبتنی بر بازپیکربندی[۳۸] حافظه نهان است که در آن بر اساس خصوصیات الگوی ورودی/خروجی بارهای کاری، پیکربندی مناسب حافظه نهانِ متناسب با آن الگو انتخاب می‌شود. ReCA شامل دو فرآیند برخط[۳۹] و برون‌خط[۴۰] است که در فرآیند برون‌خط، برنامه‌های مختلف اجرا شده و ویژگی‌های آن بارکاری تحلیل می‌شود. بارهای کاری بر اساس ویژگی‌های آن‌ها به پنج دسته تقسیم شده‌اند. سپس برای هر دسته، یک پیکربندی بهینه حافظه نهان ارائه شده است. فرآیند برخط در هنگام اجرای یک برنامه، بارکاری را بررسی کرده و تعیین می‌کند در کدام دسته قرار دارد. در صورت رخ دادن تغییر در رفتار بارکاری، پیکربندی حافظه نهان تغییر می‌کند تا با ویژگی‌های بارکاری جدید هماهنگ شود.

در [۱۰]، یک حافظه نهان آگاه از فلش و یک طرح زباله‌روبی با جابجایی صفر[۴۱] ارائه شده است. برای این منظور از خاصیت به‌روزرسانی خارج از مکان[۴۲] استفاده شده است. در این پژوهش، از داده‌های نامعتبر حاضر در دیسک حالت‌جامد، که باید طی فرایند زباله‌روبی از دیسک حالت جامد خارج شوند، استفاده شده است. این داده‌ها در یک صف نگه‌داشته می‌شوند و اگر دسترسی به آن‌ها صورت گرفت، مجدداً معتبر خواهند شد تا دسترسی به دیسک سخت کم‌تر بشود. طرح زباله‌روبی با جابجایی صفر نیز با هدف افزایش دوام دیسک حالت جامد ارائه شده که در طی زباله‌روبی، تمام داده‌های موجود در یک قطعه پاک بشوند تا سربار کپی کردن داده‌های معتبر در یک قسمت دیگر از دیسک حالت جامد کاهش بیابد.

پژوهش DEFT-Cache [۱۵] با هدف قرار دادن سامانه‌های ذخیره‌سازی مبتنی بر RAID5، در نظر دارد با استفاده از داده‌های قدیمی که منتظر فرآیند زباله‌روبی هستند ولی کماکان در حافظه نهان حضور دارند، میزان دسترسی به دیسک سخت را برای به‌روزرسانی توازن[۴۳] کاهش دهد. همچنین یک روش مبتنی بر گزارش نیز برای افزایش قابلیت اطمینان سامانه پیشنهاد شده است. در این روش، درخواست‌های نوشتن توسط یک حافظه میانگیر جمع‌آوری‌شده و به‌صورت پشت سر هم روی بخشی از دیسک سخت نوشته می‌شوند. برای مقابله با آسیب‌پذیری حافظه میانگیر در برابر قطعی برق، دوره‌ای[۴۴] برای خالی کردن حافظه میانگیر نیز در نظر گرفته شده است.

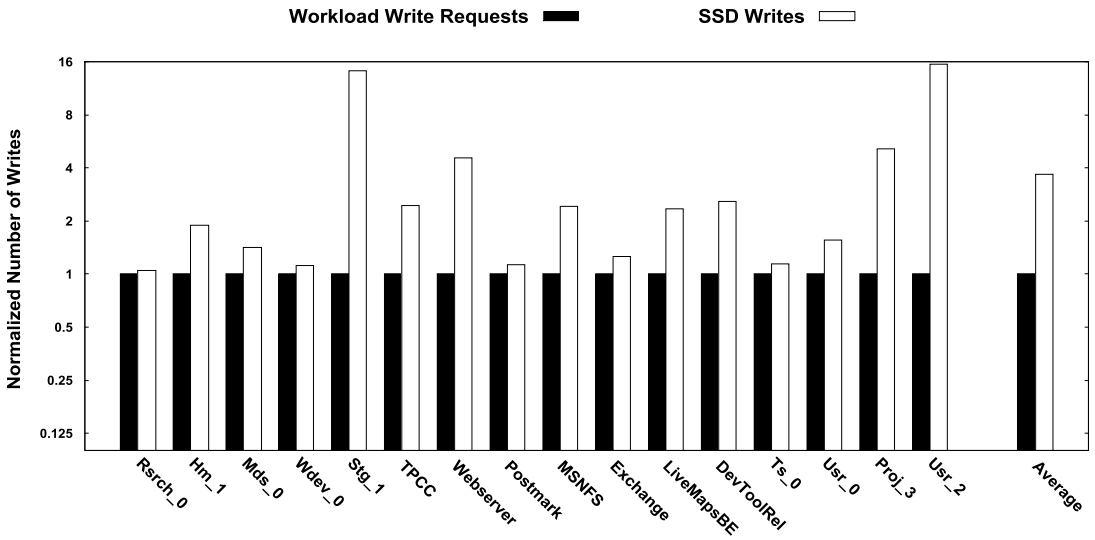
در [۴۲]، یک حافظه نهان مسطح متشکل از PCM[۴۵] و فلش پیشنهاد شده و تلاش شده تا از دوام بالاتر PCM به‌منظور بهبود طول عمر حافظه فلش استفاده بشود. در [۴۶] یک بستر کارای نهان‌سازی به‌منظور نهان‌سازی تصویر پیشنهاد شده است. در [۴۹]، یک معماری ترکیبی سه سطحی پیشنهاد شده که با استفاده از دیسک‌های حالت جامد ناهمگون[۴۶]، روشی پرهزینه به‌منظور افزایش قابلیت اطمینان حافظه نهان پس‌نویس ارائه می‌دهد. ازجمله دیگر حوزه‌های استفاده از دیسک حالت جامد به‌عنوان حافظه نهان، می‌توان به طراحی حافظه نهان ورودی/خروجی برای سامانه‌های کلید-مقدار[۴۷] اشاره کرد [۴۳][۴۴][۴۵].

۴- روش پیشنهادی

در این بخش ابتدا به طرح مسئله و بررسی چالش‌های موجود در رویکردهای مبتنی بر استفاده از دیسک حالت جامد به‌عنوان حافظه نهان می‌پردازیم. به این منظور، ابتدا به ارزیابی اثر سیاست‌های مدیریتی حافظه نهان بر دوام دیسک حالت جامد خواهیم پرداخت. سپس یک معماری دوسطحی ترکیبی به‌منظور بهبود دوام دیسک حالت جامد ارائه خواهیم داد.

۴-۱- طرح مسئله

در رویکردهای مبتنی بر استفاده از دیسک حالت جامد به‌عنوان حافظه نهان ورودی/خروجی، چگونگی مدیریت فقدان‌های خواندن ازجمله عوامل مهم و مؤثر بر عملکرد حافظه نهان است. نوشتن فقدان‌های خواندن بر روی دیسک حالت جامد، دوام دیسک حالت جامد را کاهش می‌دهد اما می‌تواند در حوزه کارایی مؤثر باشد. از طرف دیگر ننوشتن فقدان‌های خواندن بر روی دیسک حالت جامد به این معناست که این درخواست‌ها از دیسک سخت پاسخ داده شوند که باعث کاهش کارایی زیرسامانه ذخیره‌سازی خواهد شد. به‌منظور بررسی اثر نوشتن فقدان‌های خواندن بر روی دوام دیسک حالت جامد، یک شبیه‌سازی مبتنی بر ردگیری بر روی ۱۶ بارکاری انجام گرفته و تعداد دفعات نوشتن بر روی دیسک حالت جامد، استخراج شد. در این شبیه‌سازی، تعداد صفحات دیسک حالت جامد برابر با ۱۰٪ تعداد صفحات یگانه هر بارکاری قرار داده شده است.



شکل ۴- تأثیر سیاست‌های مدیریتی بر طول عمر دیسک حالت جامد

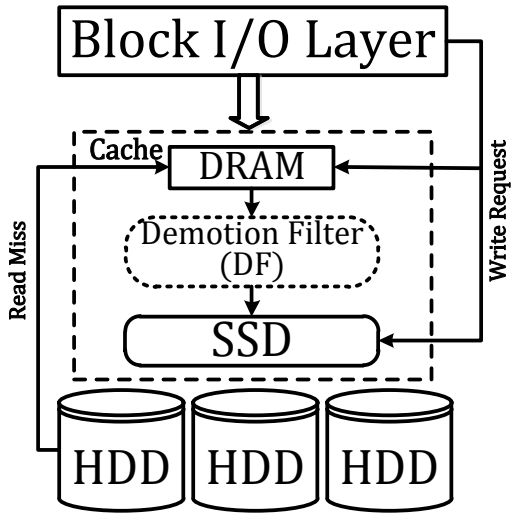
شکل ۴ تأثیر نوشتن فقدان‌های خواندن بر روی دیسک حالت جامد را در برابر نوشتن‌هایی که توسط لایه بالاتر اعمال شده، نشان می‌دهد. نوشتن فقدان‌های خواندن بر روی دیسک حالت جامد، به‌طور میانگین، منجر به ۳,۷ برابر شدن تعداد دفعات نوشتن بر روی دیسک حالت جامد می‌شود؛ افزایشی که به جهت درخواست‌های نوشتن لایه بالاتر به دیسک حالت جامد تحمیل نشده و حاصل یک سیاست مدیریتی جهت بهبود کارایی زیرسامانه ذخیره‌سازی است. استفاده از یک حافظه با دوام بالاتر از دیسک حالت جامد مبتنی بر فلش به‌عنوان حافظه نهان برای فقدان‌های خواندن، می‌تواند بدون کاهش قابلیت اطمینان سامانه، منجر به کاهش تعداد دفعات نوشتن بر روی دیسک حالت جامد شود.

۴-۲- ساختار معماری ترکیبی

فقدان‌های خواندن ازجمله مهم‌ترین عوامل مؤثر بر کارایی و دوام حافظه‌های نهان زیرسامانه‌های ذخیره‌سازی هستند. نوشتن فقدان‌های خواندن بر روی دیسک‌های حالت جامد می‌تواند منجر به کاهش طول عمر این ادوات بشود و حتی اثری بیشتر از تعداد درخواست‌های نوشتن لایه بالاتر بر روی این ادوات بگذارد. از طرفی دیگر، ننوشتن این درخواست‌ها بر روی حافظه نهان نیز می‌تواند منجر به افزایش دسترسی‌ها به دیسک سخت و کاهش کارایی بشود؛ ازاین‌رو باید به دنبال راهکارهایی جهت مدیریت فقدان‌های خواندن با در نظر گرفتن دوام و کارایی بود.

شکل ۵ ساختار کلی معماری حافظه نهان ترکیبی دوسطحی ورودی/خروجی (HyTIA) را نشان می‌دهد. در این ساختار، از DRAM به‌عنوان حافظه نهان سطح اول و از دیسک حالت جامد به‌عنوان حافظه نهان سطح دوم استفاده می‌شود. حافظه DRAM معمولاً در سطح حافظه اصلی مورد استفاده قرار می‌گیرد و استفاده از آن در زیرسامانه ذخیره‌سازی به دلیل فرار بودن این حافظه، می‌تواند منجر به کاهش قابلیت اطمینان شود. بااین‌وجود، ذخیره کردن داده‌های دست‌نخورده که از لحاظ قابلیت اطمینان بحرانی نیستند، می‌تواند منجر به کاهش تعداد دفعات نوشتن بر روی بخش‌های با دوام محدود بشود. همچنین یک فیلتر تنزل (DF) به‌منظور مهاجرت فقدان‌های خواندن از DRAM به دیسک حالت جامد پیشنهاد می‌شود. در درخواست‌های نوشتن، پیش از انجام عملیات نوشتن و درصورتی‌که آدرس مورد دسترسی از قبل در حافظه نهان حاضر باشد، آن داده در داده ساختارهای نگاشت[۴۸] نامعتبر می‌شود. سپس، درخواست نوشتن به‌صورت همزمان به DRAM و دیسک حالت جامد ارسال می‌شود. زمانی که عملیات نوشتن بر روی دیسک حالت جامد انجام شد، تصدیق[۴۹] مربوط به درخواست نوشتن به لایه بالاتر فرستاده می‌شود. به‌این‌ترتیب، اطمینان حاصل می‌شود که در صورت رخ دادن قطعی برق، داده از دست نرود.

برای درخواست‌های خواندن، بسته به حضور داده در هر کدام از اجزای حافظه نهان، مراجعه به آن جزء صورت می‌گیرد. با توجه به احتمال وجود چند نسخه از داده در اجزای مختلف حافظه نهان، در یک درخواست خواندن اولویت ابتدا با DRAM و سپس دیسک حالت جامد خواهد بود. در هنگام رخ دادن یک فقدان خواندن، داده صرفاً بر روی DRAM نوشته می‌شود. نوشتن داده بر روی DRAM به دلیل زمان پاسخ کم‌تر DRAM نسبت به دیسک سخت و دیسک حالت جامد، سربار زیادی به زمان پاسخ‌دهی به فقدان‌های خواندن اضافه نمی‌کند.



شکل ۵- ساختار کلی معماری ترکیبی دو سطحی

با روی دادن یک برخورد خواندن، دو مسئله باید مدنظر قرار گیرد. مسئله اول این است که صف‌ها و سایر داده ساختارهای مربوط به هر یک از ادوات چگونه به‌روز شوند. مسئله دوم، چگونگی مدیریت مهاجرت داده‌ها بین ادوات مختلف است. هرکدام از ادوات مورد استفاده در حافظه نهان توسط یک صف مدیریت می‌شوند و هرکدام از خانه‌های این صف شامل آدرس‌های مربوط به صفحات موجود در ادوات مورد استفاده در حافظه نهان است. همچنین این صف مطابق با سیاست نهان‌سازی، مدیریت می‌شود. در مسئله به‌روزرسانی صف‌ها و داده ساختارها، دو رویکرد را در نظر می‌گیریم:

- در رویکرد اول، با یک برخورد خواندن در یک دستگاه، صف‌های هر دو دستگاه به‌روز می‌شود. به‌این‌ترتیب داده‌های پرطرفدارتر در حافظه نهان خواهند ماند و سربار مدیریتی کم‌تری به سامانه تحمیل خواهد شد. ضعف این رویکرد، عدم استفاده مناسب از ظرفیت حافظه‌های مورد استفاده در حافظه نهان است؛ در حقیقت احتمال مراجعه به دیسک سخت توسط این رویکرد افزایش پیدا می‌کند و کارایی سامانه تحت تأثیر قرار خواهد گرفت.
- رویکرد دوم، به‌روزرسانی مستقل هر یک از ادوات است. در این روش، برخورد خواندن در هر یک از اجزای حافظه نهان، تنها منجر به به‌روز شدن صف همان جزء خواهد شد. به‌این‌ترتیب به داده‌هایی با محلیت[۵۰] کم که پرطرفدار هستند، شانس بیشتری برای حضور در حافظه نهان خواهیم داد. مشکل اصلی این رویکرد، سربار بالاتر داده ساختارهای مربوط به مدیریت داده‌ها در هر دستگاه است.

در رویکرد اول، با توجه به یکسان بودن رویکرد مدیریت در هنگام رخ دادن برخورد، این امکان وجود دارد که فضای هر دو سطح توسط یک صف مدیریت شود. به‌این‌ترتیب نیازی به صف‌های جدا برای هر سطح نیست و سطح اول حافظه نهان (که معمولاً اندازه کوچک‌تری دارد) به‌عنوان بخشی از یک صف بزرگ‌تر که سطح دوم را مدیریت می‌کند در نظر گرفته می‌شود. بااین‌وجود، این رویکرد مانع از اجرای روش‌های تنزل و ارتقا[۵۱] در حافظه نهان می‌گردد. همچنین این رویکرد شانس کم‌تری جهت برخورد صفحات فراهم می‌کند.

در معماری ترکیبی، از رویکرد دوم برای افزایش نرخ برخورد استفاده می‌شود. مسئله مدیریت مهاجرت داده‌ها ناشی از تفاوت سرعت و دوام اجزای مختلف حافظه نهان است. درواقع باید این تصمیم گرفته شود که اگر داده‌ای در دیسک حالت جامد برخورد داشت، به DRAM منتقل بشود یا خیر. با انتقال داده به DRAM، برای مجموعه‌ای از داده‌ها با محلیت زمانی، فرصت برخورد در DRAM که از دیسک حالت جامد سریع‌تر است، فراهم می‌شود. از طرفی دیگر با توجه به ظرفیت کم‌تر DRAM، داده‌های موجود در آن نسبت به داده‌های موجود در دیسک حالت جامد احتمالاً پرطرفدارتر هستند. داده‌ای که در دیسک حالت جامد برخورد خواندن داشته باشد، قطعاً در DRAM حاضر نبوده و ازاین‌رو مدتی از آخرین دسترسی به آن گذشته است؛ در نتیجه انتقال آن به DRAM ممکن است منجر به این شود که آن داده تعداد برخورد زیادی نداشته باشد و علاوه بر آن یک داده احتمالاً پرطرفدارتر نیز از DRAM خارج شود. در معماری ترکیبی، برخورد در دیسک حالت جامد منجر به مهاجرت به DRAM نمی‌شود.

چالش دیگر، تشخیص داده‌های مناسب جهت تنزل از DRAM به دیسک حالت جامد است. درج فقدان‌های خواندن در DRAM، فرصت کم‌تری به این صفحه‌ها جهت برخورد در حافظه نهان می‌دهد. از طرفی دیگر، درج صفحات بدون ارجاع در دیسک حالت جامد، تنها طول عمر حافظه نهان را کاهش می‌دهد. در این پژوهش به‌منظور تعیین صفحه‌های مناسب جهت تنزل از DRAM به دیسک حالت جامد، از یک فیلتر تنزل استفاده می‌شود. وظیفه فیلتر تنزل، جلوگیری از ورود صفحاتی با تعداد دسترسی کم به دیسک حالت جامد است. درج این صفحات در دیسک حالت جامد تنها منجر به کاهش طول عمر این ادوات می‌شود و تأثیری بر کارایی ندارد. در روش پیشنهادی، در اولین برخورد خواندن یک صفحه در DRAM، آدرس مربوط به این صفحه در یک صف به نام فیلتر تنزل درج می‌شود. درصورتی‌که در درخواست‌های بعدی، یکی از آدرس‌های مندرج در فیلتر تنزل مورد درخواست باشد، صفحه متناظر با این آدرس پرطرفدار تشخیص داده شده و در دیسک حالت جامد درج می‌شود. پس از درج در دیسک حالت جامد، آدرس از فیلتر تنزل حذف می‌گردد. فیلتر تنزل، مجموعه‌ای برابر با ۱۰۰۰ صفحه را ردگیری می‌کند.

الگوریتم ۱: روش پیشنهادی

```
Procedure HyTIA :
1   DF: Demotion Filter
2   For Each Request R
3     If R is Write Request
4       If R Exists in the Cache
5         Invalidate Previous Version of R
6       Insert R into DRAM
7       Update DRAM_Queue
8       Insert R into SSD
9       Update SSD_Queue
10    Else
11      If R Exists in the Cache
12        If R Exists in DRAM
13          Respond the Request
14          Update DRAM_Queue
15          If R Exists in DF
16            Insert  R into SSD
17            Update SSD_Queue
18            Discard the Address of R from DF
19        Else If R only Exists in DRAM
20          Insert the Address of R into DF
21      Else If R Existed in SSD
22        Respond the Request
23        Update SSD_Queue
24      Else
25        Read from HDD
26        Insert R into DRAM
27        Update DRAM_Queue
```

الگوریتم ۱، چگونگی مدیریت درخواست‌ها توسط روش پیشنهادی را نشان می‌دهد. در درخواست‌های نوشتن، در صورت وجود نسخه قدیمی داده در حافظه نهان، نامعتبر می‌گردد (خطوط ۴ و ۵ الگوریتم). سپس، داده در DRAM و SSD درج می‌شود و صف‌هایی که مدیریت داده‌ها در این دو حافظه را بر عهده دارند، به‌روز می‌شوند (خطوط ۶ تا ۹ الگوریتم). در درخواست‌های خواندن، اولویت پاسخ‌دهی (در صورت وجود داده در هرکدام از سطوح حافظه نهان) ابتدا با پاسخ از DRAM و سپس پاسخ از SSD است. در نتیجه درخواست بر اساس الویت ذکر شده پاسخ داده می‌شود و صف حافظه متناظر با پاسخ، به‌روز می‌شود (خطوط ۱۲ تا ۱۴ و ۲۱ تا ۲۳ الگوریتم). در صورت وجود داده درخواست شده در DRAM، اگر آدرس درخواست در فیلتر تنزل وجود داشته باشد، یک نسخه از داده در SSD درج می‌گردد و فراداده مربوطه از فیلتر تنزل حذف می‌گردد (خطوط ۱۵ تا ۱۸ الگوریتم). در غیر این صورت، اگر داده تنها در DRAM وجود داشته باشد، آدرس مربوط به این درخواست در فیلتر تنزل درج می‌گردد (خطوط ۱۹ و ۲۰ الگوریتم). درصورتی‌که فقدان خواندن رخ دهد، درخواست از طریق دیسک سخت پاسخ داده می‌شود و یک نسخه از داده نیز در DRAM درج می‌شود (خطوط ۲۴ تا ۲۷ الگوریتم).

۵- نتایج

در این بخش به ارزیابی معماری ترکیبی در دو حوزه دوام و کارایی خواهیم پرداخت. برای ارزیابی معماری ترکیبی، یک معماری پایه[۵۲] شامل یک دیسک حالت جامد برای مقایسه در نظر گرفته شده است. در معماری ترکیبی، DRAM و دیسک حالت جامد از سیاست LRU استفاده می‌کنند و معماری پایه نیز از سیاست LRU بهره می‌برد. به‌منظور ارزیابی، یک شبیه‌ساز مبتنی بر ردگیری توسعه داده شد که نرخ برخورد و تعداد دفعات نوشتن بر روی دیسک حالت جامد را برای هر معماری استخراج می‌کند. ارزیابی‌ها بر روی ۱۶ بارکاری رایج در حوزه ذخیره‌سازی داده انجام می‌شود [۱۶-۲۰].

در هر دو معماری، اندازه دیسک حالت جامد برابر با ۱۰٪ صفحات یگانه[۵۳] هر بارکاری قرار داده شده است. جدول ۱، مشخصات بارهای کاری استفاده شده را نشان می‌دهد. به‌منظور بررسی اثر نسبت اندازه حافظه نهان سطح اول بر روی نرخ برخورد، سه پیکربندی مختلف معماری ترکیبی (HyTIA-5، HyTIA-10 و HyTIA-15) مورد ارزیابی قرار می‌گیرد. اندازه حافظه نهان سطح اول در HyTIA-5، HyTIA-10 و HyTIA-15 به ترتیب برابر با ۵٪، ۱۰٪، و ۱۵٪ اندازه دیسک حالت جامد قرار داده شده است.

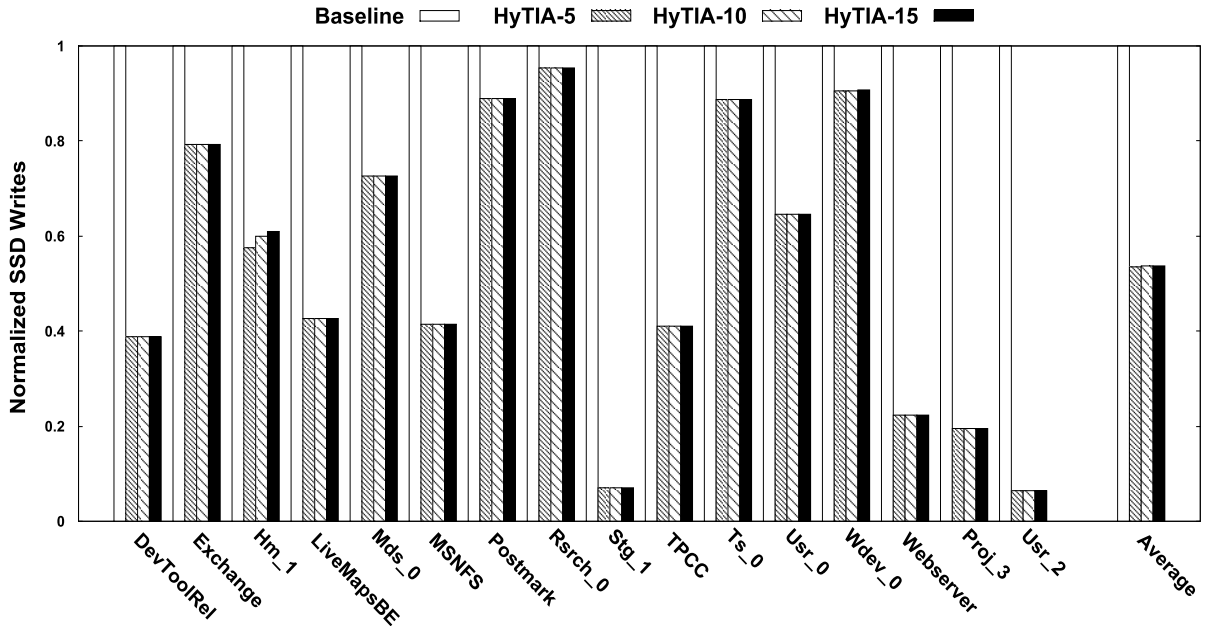
۵-۱- دوام

شکل ۷ تعداد نوشتن‌های انجام شده بر روی دیسک‌های حالت جامد را برای اندازه‌های مختلف DRAM نشان می‌دهد. نتایج معماری ترکیبی در این بخش بر نتایج معماری پایه نرمال شده است. استفاده از سیاست نوشتن فقدان‌های خواندن تنها بر روی DRAM و تنزل صفحه‌های پرطرفدارتر به دیسک حالت جامد در HyTIA-5، HyTIA-10 و HyTIA-15، به‌طور میانگین ۴۶٫۵٪، ۴۶٫۳٪ و ۴۶٫۲٪ تعداد دفعات نوشتن بر روی دیسک حالت جامد را کاهش می‌دهد. پیکربندی‌های مختلف معماری ترکیبی در بارهای کاری Usr_2، Proj_3 و Stg_1، بیش از ۸۰٪ تعداد نوشتن‌ها را کاهش داده است.

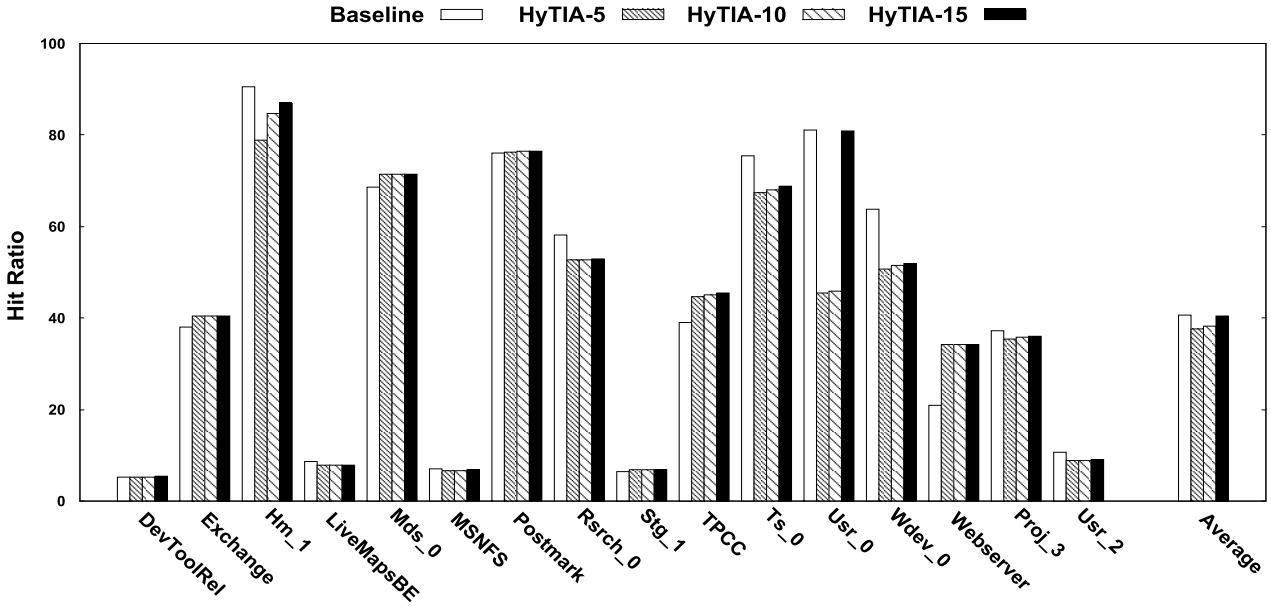
فیلتر تنزل رویکردی سخت‌گیرانه[۵۴] در انتخاب صفحه به‌منظور درج در دیسک حالت جامد دارد و همواره تعداد ثابتی از صفحه‌ها را ردگیری می‌کند. به‌این‌ترتیب مشاهده می‌شود که این رویکرد برای تمام بارهای کاری، به‌جز Hm_1، عملکرد تقریباً ثابتی (حداکثر ۰٫۲٪ اختلاف) در حوزه دوام برای پیکربندی‌های مختلف HyTIA ارائه می‌دهد. در بارکاری Hm_1، اندازه بزرگ‌تر حافظه نهان سطح اول موجب می‌شود تا هر فقدان خواندن، فرصت بیشتری جهت برخورد و در نتیجه تنزل توسط DF در اختیار داشته باشد. به‌این‌ترتیب، با افزایش اندازه حافظه نهان سطح اول، تعداد دفعات نوشتن بر روی دیسک حالت جامد نیز افزایش می‌یابد. این در حالی است که در سایر بارهای کاری، DF تقریباً میزان یکسانی صفحه را تنزل داده است. به‌این‌ترتیب HyTIA، در کنار نوشتن‌های ارسالی توسط لایه بالاتر، تقریباً تعداد یکسانی از صفحه‌های پرطرفدار را تنزل داده است. در نتیجه می‌توان با استفاده از HyTIA-5، با صرف هزینه کم‌تر، عملکرد یکسانی در حوزه دوام نسبت به پیکربندی‌های دیگر به دست آورد. علی‌رغم بهبودی که معماری ترکیبی در بارهای کاری با تعداد درخواست‌های خواندن زیاد ارائه می‌دهد، این معماری نیاز به بهینه‌سازی در حوزه بارهای کاری عمدتاً نوشتنی[۵۵] دارد. به دلیل آسیب‌پذیری DRAM در برابر قطعی برق، این حافظه گزینه مطلوبی برای ذخیره کردن درخواست‌های نوشتن نیست و باید یک نسخه از درخواست نوشتن به ادوات غیرفرار ارسال بشود. استفاده از حافظه‌های غیرفرار نوظهور مانند STT-RAM[۵۶] [۲۷][۲۸] و PCM [۲۹][۳۰] به‌عنوان حافظه نهان سطح اول، امکان بهینه‌سازی برای بارهای کاری عمدتاً نوشتنی را فراهم می‌کند.

جدول ۱- مشخصات بارهای کاری

بارکاری	درخواست‌های نوشتن	درخواست‌های خواندن	اندازه (GB)
TPCC	٪۳۰	٪۷۰	۴۳,۹۳۲
Webserver	٪۳۹	٪۶۱	۶,۶۰۷
DevToolRel	٪۳۲	٪۶۸	۳,۱۳۳
LiveMapsBE	٪۲۹	٪۷۱	۱۵,۶۴۶
MSNFS	٪۳۵	٪۶۵	۱۰,۲۵۱
Exchange	٪۷۶	٪۲۴	۹,۷۹۵
Postmark	٪۷۱	٪۲۹	۱۹,۴۳۷
Stg_1	٪۳۶	٪۶۴	۹۱,۸۱۵
Rsrch_0	٪۹۱	٪۹	۱۳,۱۱
Wdev_0	٪۸۰	٪۲۰	۱۰,۶۲۸
Ts_0	٪۸۲	٪۱۸	۱۶,۶۱۲
Usr_0	٪۶۰	٪۴۰	۵۱,۹۴۵
Hm_1	٪۶	٪۹۴	۹,۴۵
Mds_0	٪۶۹	٪۳۱	۱۱,۴
Proj_3	٪۵	٪۹۵	۲۰,۸۶۴
Usr_2	٪۱۹	٪۸۱	۴۴۱,۷۵۲



شکل ۷- تعداد نوشتن‌های انجام شده بر روی دیسک حالت جامد



شکل ۸- نرخ برخورد

۵-۲- نرخ برخورد

از دیگر ویژگی‌های مهم یک معماری کارا، عملکرد آن معماری در حوزه کارایی است. علی‌رغم عملکرد مناسب معماری ترکیبی در حوزه دوام، رویکرد درج نکردن فقدان-های خواندن در سطح دوم حافظه نهان می‌تواند منجر به افزایش دسترسی‌ها به دیسک سخت و کاهش کارایی زیرسامانه ذخیره‌سازی شود. با بررسی نرخ برخورد، می‌توانیم به تصویری کلی از چگونگی عملکرد هر معماری در حوزه کارایی برسیم.

شکل ۸ نرخ برخورد سه پیکربندی مختلف HyTIA و معماری پایه را نشان می‌دهد. به‌طور میانگین، معماری پایه نرخ برخوردی برابر با ۴۲٫۹٪ ارائه می‌دهد. همچنین، پیکربندی‌های مختلف معماری ترکیبی (HyTIA-5، HyTIA-10 و HyTIA-15) به ترتیب نرخ برخورد میانگین برابر با ۳۹٫۵٪، ۴۰٫۱٪ و ۴۲٫۶٪ ارائه می‌دهند. نرخ برخورد بالاتر معماری پایه به سبب درج فقدان‌های خواندن در دیسک حالت جامد است که فرصت بیشتری در اختیار داده‌ها قرار می‌دهد. در بارهای کاری Usr_2، Ts_0 و Rsrch_0، افزایش اندازه حافظه نهان سطح اول نیز نتوانسته منجر به افزایش نرخ برخورد شود. در این بارهای کاری، فاصله استفاده مجدد[۵۷] فقدان‌های خواندن بیشتر از اندازه حافظه نهان سطح اول است. به‌منظور بهبود کارایی، باید نرخ تنزل داده‌ها از سطح اول به سطح دوم را افزایش داد تا به‌این‌ترتیب، فرصت بیشتری در اختیار صفحات قرار بگیرد.

نکته دیگری که باید به آن اشاره کرد، تأثیر افزایش اندازه حافظه نهان سطح اول است. به‌جز بارهای کاری Hm_1 و Usr_0، دو پیکربندی HyTIA-5 و HyTIA-15 در سایر بارهای کاری نرخ برخورد تقریباً یکسانی ارائه می‌دهند. این در حالی است که قیمت هر گیگابایت DRAM می‌تواند ده برابر هر گیگابایت دیسک حالت جامد مبتنی بر فلش باشد [۲۴][۲۵]. به‌این‌ترتیب HyTIA-5 با ۴۰٪ هزینه کم‌تر نسبت به HyTIA-15، در ۱۴ بارکاری نرخ برخورد تقریباً یکسانی ارائه می‌دهد و یک پیکربندی کارا از نظر هزینه به شمار می‌آید. استفاده از رویکردهای ساده‌تر تنزل صفحه‌ها در Hm_1 و Usr_0 می‌تواند نرخ برخورد میانگین HyTIA-5 را نیز بهبود دهد.

۶- جمع‌بندی و کارهای آتی

در این مقاله، تأثیر معماری ترکیبی دو سطحی بر روی دوام و کارایی حافظه نهان ورودی/خروجی مورد ارزیابی قرار گرفت. در معماری ترکیبی، از یک حافظه با دوام بالا (مانند DRAM) به‌عنوان سطح اول و از دیسک حالت جامد به‌عنوان سطح دوم حافظه نهان استفاده و تلاش شد تا با تحمیل بار بیشتر بر روی سطح اول حافظه نهان، طول عمر دیسک حالت جامد افزایش یابد. در ساختار معماری ترکیبی پیشنهادی، فقدان‌های خواندن تنها در سطح اول حافظه نهان درج می‌شوند تا تعداد دفعات نوشتن بر روی دیسک حالت جامد کاهش یابد. همچنین یک فیلتر تنزل به‌منظور تشخیص داده‌های مناسب جهت درج فقدان‌های خواندن در دیسک حالت جامد پیشنهاد شد. ارزیابی‌های انجام شده نشان داد معماری ترکیبی می‌تواند نسبت به معماری پایه، به‌طور میانگین ۴۶٫۵٪ تعداد دفعات نوشتن بر روی دیسک حالت جامد را کاهش دهد.

در کارهای آتی می‌توان روش‌هایی برای ارتقای صفحات از دیسک حالت جامد به DRAM به‌منظور افزایش کارایی حافظه نهان ارائه داد. استفاده از رویکردهای پویا در DF نیز ازجمله روش‌هایی است که می‌تواند منجر به بهبود کارایی بشود. همچنین می‌توان به بررسی اثر استفاده از حافظه‌های غیرفرار نوظهور به‌عنوان سطح اول حافظه نهان ورودی/خروجی پرداخت. ارائه مدل هزینه-فایده به‌منظور ارزیابی جامع-تر معماری ترکیبی نیز ازجمله رویکردهایی است که می‌توان در کارهای آتی در پیش گرفت.

۷- مراجع

[1] M. Khoso, "How Much Data Is Produced Every Day?," [Online].Available:http://www.northeastern.edu/levelblog/2016/05/13/how-much-data-produced-every-day/. Accessed:1/01/2018.

[2] F. Chen, D. A. Koufaty, and X. Zhang, "Understanding Intrinsic Characteristics and System Implications of Flash Memory Based Solid State Drives," *32th International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS),* New York, NY, USA, pp. 181-192, Aug. 2009.

[3] Y. Ni, J. Jiang, D. Jiang, X. Ma, J. Xiong, and Y. Wang, "S-RAC: SSD Friendly Caching for Data Center Workloads," *9th ACM International System Storage Conference (SYSTOR),* pp. 8:1-8:12, Jun. 2016.

[4] Y. Liang, Y. Chai, N. Bao, H. Chen, and Y Liu, "Elastic Queue: A Universal SSD Lifetime Extension Plug-in for Cache Replacement Algorithms," *9th ACM International Systems & Storage Conference (SYSTOR),* pp. 5:1-5:11, Jun. 2016.

[5] J. Kim, S. Seo, D. Jung, J. S. Kim, and J. Huh, "Parameter-Aware I/OManagement for Solid State Disks (SSDs)," *IEEE Transactions on Computers(TC)*, vol. 61, no. 5, pp. 636 –649, May 2012.

[6] R. Micheloni, A. Marelli, and K. Eshghi, "Inside Solid State Drives (SSDs)," *Springer Series in Advanced Microelectronics,* 2018.

[7] R. Salkhordeh, S. Ebrahimi, and H. Asadi, "ReCA: an Efficient Reconfigurable Cache Architecture for Storage Systems with Online Workload Characterization," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 29, no.7, pp.1605-1620, Jul. 2018.

[8] S. Huang, Q. Wei, J. Chen, C. Chen, and D. Feng, "Improving Flash-Based Disk Cache with Lazy Adaptive Replacement," *29th IEEE Symposium on Mass Storage Systems and Technologies (MSST)*, Long Beach, CA, USA, pp. 1-10, May 2013.

[9] R. Santana, S. Lyons, R. Koller, R. Rangaswami, and J. Liu, "To ARC or not to ARC," *7th USENIX Conference on Hot Topics in Storage and File Systems (HotStorage)*, Santa Clara, CA, USA, pp. 14–14, Jul. 2015.

[10] Q. Xia and W. Xiao, "High-Performance and Endurable Cache Management for Flash-Based Read Caching," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 27, no. 12, pp. 3518-3531, Dec. 2016.

[11] D. Jiang, Y. Che, J. Xiong, and X. Ma, "uCache: A Utility-Aware Multilevel SSD Cache Management Policy," *10th IEEE International Conference on High Performance Computing and Communications & IEEE International Conference on Embedded and Ubiquitous Computing (HPCC_EUC)*, Zhangjiajie, China, pp. 391-398, Nov. 2013.

[12] Z. Chen, N. Xiao, Y. Lu, and F. Liu, "Me-CLOCK:A Memory-Efficient Framework to Implement Replacement Policies for Large Caches," *IEEE Transactions on Computers (TC)*, vol. 65, no. 8, pp. 2665-2671, Aug. 2016.

[13] Y. Chai, Z. Du, X. Qin, and D. A. Bader, "WEC: Improving Durability of SSD Cache Drives by Caching Write- EfficientData," *IEEE Transactions on Computers (TC)*, vol. 64, no. 11, pp. 3304-3316, Nov. 2015.

[14] S. He, Y. Wang, and X. H. Sun, "Improving Performance of Parallel I/O Systems through Selective and Layout-Aware SSD Cache," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 27, no. 10, pp. 2940- 2952, Oct. 2016.

[15] J. Wan, W. Wu, L. Zhan, Q. Yang, X. Qu, and C. Xie, "DEFT-Cache: A Cost-Effective and Highly Reliable SSD Cache for RAID Storage," *IEEE International Parallel and Distributed Processing Symposium (IPDPS),* Orlando, FL, USA, pp. 102-111, 2017.

[16] D. Narayanan, A. Donnelly, and A. Rowstron, "Write Off-Loading: Practical Power Management for Enterprise Storage,"*ACM Transactions on Storage (TOS)*, vol. 4, no. 3, pp. 1–23, Nov. 2008.

[17] Storage Networking Industry Association IOTTA Repository, "Microsoft Enterprise Traces," [Online]. Available: http://iotta.snia.org/traces/130. Accessed: 01/01/2018.

[18] Storage Networking Industry Association IOTTA Repository, "Microsoft Production Server Traces," [Online]. Available: http://iotta.snia.org/traces/158. Accessed: 01/01/2018.

[19] J. Katcher, "Postmark: A New File System Bench-Mark," *Network Appliance, Technical Report*. 3022, Oct. 1997.

[20] A. Wilson, "The New and Improved FileBench," *6th USENIX Conference on File and Storage Technologies (FAST),* San Jose, CA, USA, Feb. 2008.

[21] Intel, "Game-changing Fast Memory Also a Big Deal for Big Data," [Online]. Available: https://newsroom.intel.com/e di torials/ 3d-xpoint-memory-storage.Accessed: 23/06/2018.

[22] F. Chen, D. A. Koufaty, and X. Zhang, "Hystor: Making the Best Use of Solid State Drives in High Performance Storage Systems," *25th International Conference on Supercomputing (ICS)*, Tucson, AR., USA, pp. 22–32, Jun. 2011.

[23] S. K. Yoon, Y. S. Youn, S. J. Nam, M. H. Son, and S. D. Kim, "Optimized Memory-Disk Integrated System with DRAM and Nonvolatile Memory," *IEEE Transactions on Multi-Scale Computing Systems (TMSCS)*, vol. 2, no. 2, pp. 83- 93, Apr. 2016.

[24] Samsung, "SAMSUNG Memory/Storage MSRP Price List-EFFECTIVE April 2017," *Technical Report.* 2017.

[25] PCPARTPICKER, "GeIL - EVO Veloce Series 8GB (2x4GB) DDR3-1600 Memory,"[Online]. Available:https://pcpartpicker.com/ product/9nzv6h/geil-memorygev38gb1600c9dc? history_days= 730. Accessed: 23/06/2018.

[26] W. Fischer, "Linux Storage Stack Diagram," [Online]. Available: https://www.thomaskrenn.com/en/wiki/Linux_Storage_Stack_ Diagram. Accessed: 08/04/2019.

[27] E. Cheshmikhani, H. Farbeh, and H. Asadi, "Enhancing Reliability of STT-MRAM Caches by Eliminating Read Disturbance Accumulation," *54th IEEE/ACM Design, Automation and Test in Europe Conference (DATE)*, Florence, Italy, Mar. 2019.

[28] E. Cheshmikhani, H. Farbeh, and H. Asadi, "ROBIN: Incremental Oblique Interleaved ECC for Reliability Improvement in STT-MRAM Caches," *24th IEEE Asia and South Pacific Design Automation Conference (ASP-DAC),* Tokyo, Japan, Jan. 2019.

[29] R. Salkhordeh and H. Asadi, "An Operating System level data migration scheme in hybrid DRAM-NVM memory architecture," *51st IEEE/ACM Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Dresden, Germany, pp. 936-941, Mar. 2016.

[30] M. Bazzaz, A. Hoseinghorban, F. Poursafaei, and A. Ejlali, "High-Performance Predictable NVM-based Instruction Memory for Real-Time Embedded Systems," *IEEE Transactions on Emerging Topics in Computing (TETC)*, Early Access, pp. 1-1, Jul. 2018.

[31] Windows Central, "Dealing with RPM: Why laptops are still using slow hard drives," [Online]. Available: https://www.windowscentral.com/dealing-rpm-why-laptops-are-still -using-slow-hard-drives. Accessed: 09/04/2019.

[32] Samsung, "Samsung SM863a SSD for Data Centers | Enterprise SSD | Samsung Semiconductor Global Website," [Online]. Available : https://www.samsung.com/semiconductor/minisite/ssd/enterprise/sm863a. Accessed: 22/06/2018.

[33] M. Rouse, "PCIe SSD (PCIe solid-state drive)," [Online]. Available: https://searchstorage.techtarget.com/definition/PCIe-SSD-PCIe-solid-state-drive. Accessed: 09/04/2019.

[34] S. He, Y. Wang, Z. Li, X. H. Sun, and C. Xu, "Cost-Aware Region-Level Data Placement in Multi-Tiered Parallel I/O Systems," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 28, no. 7, pp. 1853-1865, Jul. 2017.

[35] R. Salkhordeh, H. Asadi, and S. Ebrahimi, "Operating System Level Data Tiering Using Online Workload Characterization," *The Journal of Supercomputing*, vol. 71, no. 4, pp. 1534-1562, Apr. 2015.

[36] D. Lee, S. Yoon, J. Kim, C. C. Weems, and S. D. Kim, "A New Memory-Disk Integrated System with HW Optimizer*," ACM Transactions on Architecture and Code Optimization (TACO)*, vol. 12, no. 2, pp. 11:1-11:23, Apr. 2015.

[37] S. K. Yoon, M. Y. Bian, and S. D. Kim, "An Integrated Memory-Disk System with Buffering Adapter and Non-Volatile Memory," *Springer Design Automation for Embedded System*, vol. 17, no. 3, pp. 609-626, Sep. 2013.

[38] J. Cui, Y. Zhang, J. Huang, W. Wu, and J. Yang, "ShadowGC: Cooperative Garbage Collection with Multi-Level Buffer for Performance Improvement in NAND Flash-Based SSDs," *53rd IEEE/ACM Design, Automation & Test in Europe Conference & Exhibition (DATE)*, Dresden, Germany, pp. 1247-1252, Mar. 2018.

[39] M. Tarihi, H. Asadi, A. Haghdoost, M. Arjomand, and H. Sarbazi-Azad, "A Hybrid Non-Volatile Cache Design for Solid-State Drives Using Comprehensive I/O Characterization," *IEEE Transactions on Computers (TC)*, vol. 65, no. 6, pp. 1678-1691, Jun. 2016.

[40] E. Lee, J. Kim, H. Bahn, S. Lee, and S. H. Noh, "Reducing Write Amplification of Flash Storage through Cooperative Data Management with NVM," *ACM Transactions on Storage (TOS)*, vol. 13, no. 2, pp. 12:1--12:13, Jun. 2017.

[41] D. H. Kang, S. J. Han, Y. C. Kim, and Y. I. Eom, "CLOCK-DNV: A Write Buffer Algorithm for Flash Storage Devices of Consumer Electronics," *IEEE Transactions on Consumer Electronics (TCE)*, vol. 63, no. 1, pp. 85-91, Feb. 2017.

[42] L. Shi, J. Li, C. J. Xue, and X. Zhou, "Hybrid Nonvolatile Disk Cache for Energy-efficient and High-performance Systems," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 18, no. 1, pp. 8:1-8:23, Jan. 2013.

[43] Z. Shen, F. Chen, Y. Jia, and Z. Shao, "DIDACache: A Deep Integration of Device and Application for Flash Based Key-Value Caching," *15th USENIX Conference on File and Storage Technologies (FAST)*, Santa Clara, CA, USA, pp. 391-405, Feb. 2017.

[44] S. Xu, S. Lee, S. W. Jun, M. Liu, and J. Hicks, "Bluecache: A Scalable Distributed Flash-Based Key-Value Store," *42nd international conference on very large data bases (VLDB)*, New Delhi, India, pp. 301-312, Mar. 2016.

[45] Z. Shen, F. Chen, Y. Jia, and Z. Shao, "Optimizing Flash-based Key-value Cache Systems," *8th USENIX Conference on Hot Topics in Storage and File Systems (HotStorage)*, Denver, CO, USA, pp. 1-5, Jun. 2016.

[46] L. Tang, Q. Huang, W. Lloyd, S. Kumar, and K. Li, "RIPQ: Advanced Photo Caching on Flash for Facebook," *13th USENIX Conference on File and Storage Technologies (FAST)*, Santa Clara, CA, USA, pp. 373-386, Feb. 2015.

[47] J. Niu, J. Xu, and L. Xie, "Hybrid Storage Systems: A Survey of Architectures and Algorithms," *IEEE Access*, vol. 6, pp. 13385-13406, Feb. 2018.
[48] Seagate, "Mobile SATA Momentus," [Online.] Available: https://www.seagate.com/files/staticfiles/support/docs/samsung-ds/ 100698122c.pdf. Accessed: 30/06/2018.
[49] R. Salkhordeh, M. Hadizadeh, and H. Asadi, "An Efficient Hybrid I/O Caching Architecture Using Heterogeneous SSDs," *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, Early Access, pp. 1-1, Nov. 2018.

مصطفی هادی‌زاده مدرک کارشناسی مهندسی کامپیوتر خود را از دانشگاه شهید بهشتی در سال ۱۳۹۵ و مدرک کارشناسی ارشد معماری کامپیوتر خود را از دانشگاه صنعتی شریف در سال ۱۳۹۷ دریافت نمود. زمینه‌های تحقیقاتی موردعلاقه وی شامل معماری کامپیوتر، سامانه‌های حافظه و ذخیره‌سازی داده ، سامانه‌های روی تراشه و سامانه‌های اتکاپذیر می‌باشد. ایشان هم‌اکنون به‌عنوان دستیار پژوهشی در آزمایشگاه ذخیره‌سازی، پردازش و شبکه‌های داده (DSN) در دانشگاه صنعتی شریف مشغول می‌باشد. آدرس پست الکترونیکی ایشان عبارت است از:
mhadizadeh@ce.sharif.edu

رضا سالخورده حقیقی مدرک کارشناسی خود را از دانشگاه فردوسی مشهد دریافت کرده است. ایشان مدرک کارشناسی ارشد و دکتری خود را به ترتیب در سال‌های ۲۰۱۳ و ۲۰۱۸ اخذ نموده است. علایق تحقیقاتی ایشان شامل دیسک‌های حالت جامد، سامانه‌های ذخیره‌سازی داده، حافظه‌های غیرفرار و سیستم‌عامل می‌باشد. ایشان هم‌اکنون به‌عنوان محقق پسادکتری در دانشگاه واترلو مشغول می‌باشد. آدرس پست الکترونیکی ایشان عبارت است از:
reza.salkhordeh@sharif.edu

حسین اسدی استاد دانشکده مهندسی کامپیوتر دانشگاه صنعتی شریف است .ایشان بنیان‌گذار و مدیر آزمایشگاه ذخیره‌سازی، پردازش و شبکه‌های داده (DSN) می‌باشد که در طی چند سال اخیر بیش از ۵۰ دانشجوی کارشناسی، کارشناسی ارشد و دکترا پروژه خود را در این آزمایشگاه با موفقیت انجام داده‌اند. در سال ۲۰۱۱، او جایزه استاد نمونه دانشکده مهندسی کامپیوتر را به خود اختصاص داد. دکتر اسدی در سال ۱۳۹۲ به درجه دانشیاری ارتقاء یافت. ایشان در سال ۱۳۹۴، اولین شرکت طراحی و تولیدکننده محصولات ذخیره‌سازی با برند HPDS را تأسیس نمود. در سال ۱۳۹۵، دو جایزه معتبر "محقق برجسته" و "جایزه موسسه پژوهشی برجسته" از جانب دانشگاه صنعتی شریف به ایشان اعطاء گردید. در سال ۱۳۹۶ نیز "جایزه فناور برجسته" توسط دانشگاه به ایشان اعطاء گردید. پیش از پیوستن به دانشگاه صنعتی شریف، ایشان به مدت سه سال در شرکت EMC به‌عنوان محقق و مهندس ارشد مشغول به کار بود. ایشان بیش از هشتاد مقاله در نشریات مشهور و مقالات کنفرانس، نویسنده یا همکار نویسنده بوده است. او همچنین به‌عنوان دبیر مهمان IEEE Transactions on Computers، دبیر مجله Elservier Microlectronics Reliability خدمت کرده است. اخیراً ایشان به‌عنوان استاد مدعو از دانشگاه EPFL بازدید نموده است. ایشان عضو ارشد IEEE است. آدرس پست الکترونیکی ایشان عبارت است از:
asadi@sharif.edu

[1] Reliability
[2] Performance
[3] Solid-State Drive
[4] Devices
[5] Endurance
[6] Non-volatile
[7] Dirty
[8] Demotion Filter
[9] Trace-based Simulator
[10] Hybrid Two-Level I/O Cache Architecture
[11] Clean
[12] Seek Time
[13] Rotation
[14] Die
[15] Plane
[16] Block
[17] Page
[18] Garbage Collection
[19] Wear Leveling
[20] Tiering
[21] Memory-Disk Integrated Systems
[22] Input/Output
[23] Device
[24] Write-Through
[25] Write-Back
[26] Lazy
[27] Least Recntly Used
[28] Working Set
[29] State
[30] Caching
[31] Bloom Filter
[32] Tracking
[33] Write-efficient Caching
[34] Pulling
[35] Proactive
[36] Selective
[37] Layout
[38] Reconfiguration
[39] Online
[40] Offline
[41] Zero-migration
[42] Out of Place Update
[43] Parity
[44] Period
[45] Phase-Change Memory
[46] Heterogeneous
[47] Key-Value
[48] Mapping
[49] Acknowledgement
[50] Locality
[51] Promotion
[52] Baseline
[53] Unique
[54] Strict
[55] Write-Intensive
[56] Spin-Transfer Torque Magnetic Random-Access Memory
[57] Reuse Distance

The CSI Journal on Computing Science and Information Technology
Vol. 17, No.1, 2019

Enhancing I/O Cache Lifetime Using Hybrid Multi-Level Architecture

Mostafa Hadizadeh, Reza Salkhordeh, Hossein Asadi

Department of Computer Engineering, Sharif University of Technology, Tehran, Iran.

Abstract

Employing Solid-State Drives (SSDs) as a cache in data storage subsystem is a cost-efficient approach for designing high-performance systems. This approach, however, suffers from limited lifetime of SSDs. Based on our evaluations and through comprehensive investigations, we show that one of the major sources of shortening SSD lifetime is placing data pages in the SSD in case of read misses. Although such policy improves the hit ratio, it can impose up to 15.6x more writes to the SSD. In this paper, we present Hybrid Two-Level I/O Cahce Architecture (HyTIA) to enhance the SSD lifetime. HyTIA employs DRAM as the first level and SSD as the second level cache. Additionally, we propose a Demotion Filter (DF) to reduce the admission rate of the missed read data pages to SSD with negligible performance overhead (0.3%, on average). Our evaluations show that, on average, HyTIA reduces the number of SSD writes by 46.5% (up to 93%).

Keywords: Data Storage Subsystem, Solid-State Drive, Hybrid Multi-Level Cache.