Factores que afectan el consumo energético de operaciones de checkpoint y restart en clusters

Marina Morán[1], Javier Balladini[1], Dolores Rexachs[2], Emilio Luque[2]

[1]Facultad de Informática, Universidad Nacional del Comahue
Buenos Aires 1400, 8300 Neuquén, Argentina
{javier.balladini, marina}@fi.uncoma.edu.ar
[2]Departamento de Arquitectura de Computadores y Sistemas Operativos, Universitat Autónoma de Barcelona
Campus UAB, Edifici Q, 08193 Bellaterra (Barcelona), España
{dolores.rexachs, emilio.luque}@uab.es

Resumen El método de tolerancia a fallos mas usado actualmente en Cómputo de Altas Prestaciones es el de rollback-recovery mediante el uso de checkpoints. Éste, como cualquier otro método de tolerancia a fallos, agrega un consumo energético adicional al propio de la ejecución de la aplicación. El objetivo de este trabajo es determinar los factores que afectan el consumo energético de los nodos de cómputo de un cluster homogéneo, al ejecutar operaciones de checkpoint y restart, sobre aplicaciones SPMD (Single Program Multiple Data). Nos hemos enfocado en el estudio energético de nodos de cómputo, contemplando diferentes configuraciones de parámetros de hardware y software. Se estudió el efecto de los estados de rendimiento (estados P) y potencia (estados C) de los procesadores, el tamaño del problema de la aplicación, la configuración del software de checkpoint utilizado (DMTCP), y del sistema de archivos distribuído (NFS). El análisis de los resultados permitió identificar oportunidades que permiten disminuir el consumo energético de las operaciones de checkpoint y restart.

1. Introducción

La computación de altas prestaciones (HPC, High Performance Computing) continúa aumentando su poder de cómputo al mismo tiempo que aumenta su consumo energético. Dadas las limitaciones que existen para abastecer de energía a este tipo de computadoras, se hace necesario conocer el comportamiento del consumo energético en estos sistemas, y encontrar formas de disminuirlo o acotarlo. En especial, para la era exaescala, se estima un límite máximo de 20 MW [4].

El método de tolerancia a fallos mas usado actualmente en CAP es el de rollback-recovery mediante el uso de checkpoints. Éste, como cualquier otro método de tolerancia a fallos, agrega un consumo energético adicional al propio de la ejecución de la aplicación [16]. Debido a ello, es importante conocer y predecir el comportamiento energético de métodos de tolerancia a fallos, de manera de

poder gestionar su impacto en el consumo total de energía durante la ejecución de una aplicación.

Un sistema de cómputo de tipo cluster tiene nodos de cómputo, nodos de almacenamiento, y al menos una red de interconexión. El objetivo de este trabajo es determinar los factores que afectan el consumo energético de los nodos de cómputo de un cluster homogéneo, al ejecutar operaciones de checkpoint y restart (C/R), sobre aplicaciones SPMD (Single Program Multiple Data). Nos hemos enfocado en el estudio energético de nodos de cómputo, contemplando diferentes configuraciones de parámetros de hardware y software. El consumo energético del nodo de almacenamiento y la red de almacenamiento no se consideran en este artículo.

Las contribuciones de este artículo son:

- Un estudio de factores propios del sistema (hardware y software) y de las aplicaciones que impactan en el consumo energético producido por las operaciones de checkpoint y restart.
- La identificación de oportunidades que permitan disminuir el consumo energético de las operaciones de checkpoint y restart.

La sección 2 presenta algunos trabajos relacionados, mientras que en la sección 3 se identifican los factores que inciden en el consumo energético. En la sección 4 se detalla la plataforma experimental y el diseño de experimentos, cuyos resultados y su análisis son presentados en la sección 5. Finalmente, se presentan las conclusiones y trabajos futuros en la sección 6.

2. Trabajos Relacionados

En [8] y [11] evalúan el comportamiento energético del C/R coordinado y no coordinado con logeo de mensajes. En [11] además evalúan la recuperación paralela y proponen un modelo analítico para predecir el comportamiento de dichos protocolos a escala exa. En [12] utilizan un modelo analítico para comparar el tiempo de ejecución y la energía consumida de la replicación y del C/R coordinado. En [2] y [6] presentan un modelo analítico para estimar el intervalo óptimo de un checkpoint multinivel en términos de consumo energético. No miden potencia disipada sino que utilizan valores de otras publicaciones. En [7] miden la potencia disipada y el tiempo de ejecución de las operaciones de alto nivel involucradas en el checkpoint (coordinados, no coordinados y jerárquicos) variando el número de núcleos involucrados. No utilizan diferentes frecuencias de procesador, ni indican si el checkpoint es comprimido o no. En [14], [5] y [15] presentan un marco para el ahorro energético del C/R. En [14] reemplazan muchas operaciones de E/S pequeñas por pocas grandes a cargo de un solo núcleo, para hacer más eficiente energéticamente el checkpoint y el restart. Utilizan RAPL para medir y limitar el consumo energético. En [15] diseñan un *runtime* que permite modificar la frecuencia de reloj y el número de procesos que realizan las operaciones de E/S del C/R de manera de optimizar el consumo energético. Otro trabajo donde se analiza el impacto de del escalado dinámico de frecuencia

y tensión en el consumo energético de las operaciones de checkpoint es en [13]. En [9] evalúan el consumo energético de una aplicación que utiliza checkpoints comprimidos. Muestran que al utilizar compresión se gasta más energía pero se ahorra tiempo, por lo que la ejecución completa de la aplicación con todos sus checkpoints puede verse beneficiada desde el punto de vista energético.

Nuestro trabajo se centra en C/R coordinado a nivel de sistema. Los valores de potencia disipada de las operaciones de checkpoint y restart son mediciones obtenidas con un medidor físico externo. No hemos encontrado trabajos que evalúen el impacto de los estados C y de las configuraciones del NFS en el consumo energético de las operaciones de C/R.

3. Identificación de factores que inciden en el consumo energético

La energía se puede calcular como el producto entre la potencia y el tiempo. Cualquier factor que pueda incidir sobre alguno de estos dos parámetros debe ser contemplado para luego analizar de qué manera afecta al consumo energético. Enfocándonos en el estudio energético de nodos de cómputo, los factores están relacionados a diferentes niveles: Hardware, Software de Aplicación y Software de Sistema.

Hardware: La especificación de Interfaz de Potencia y Configuración Avanzada (ACPI, acrónimo de Advanced Configuration and Power Interface) provee un estandar abierto que permite al sistema operativo gestionar la energía de los dispositivos y del sistema de cómputo entero [1]. Permite gestionar el comportamiento energético del procesador, el componente que más energía consume en un sistema de cómputo. ACPI define Estados de Potencia del Procesador (estados Cx), donde C0 es el estado de ejecución, y C1... Cx son estados de inactividad.

Un procesador que está en estado C0, también estará en un Estado de Rendimiento (estados Px). El estado P0 significa una ejecución a la máxima capacidad de rendimiento y demanda de potencia. A medida que se aumenta el número del estado P, se reduce su rendimiento y potencia demandada. Los procesadores implementan los estados P utilizando la técnica de Escalado Dinámico de Frecuencia y Tensión (DVFS, acrónimo de Dynamic Voltage and Frequency Scaling) [10]. Reduciendo la tensión suministrada se reduce el consumo energético. Sin embargo, se incrementa el retardo de las compuertas lógicas, entonces es necesario reducir la frecuencia de reloj de la CPU para que el circuito funcione correctamente. En ciertos procesadores multicore, se permite que cada núcleo esté en un estado P diferente. Normalmente, la técnica DVFS logra mejoras cuando una carga de trabajo está limitada por memoria, es decir, el procesador malgasta muchos ciclos de CPU esperando obtener datos desde la memoria.

Cuando no hay instrucciones para ejecutar, el procesador puede ser puesto en un estado C mayor a 0, para ahorrar energía. Existen diferentes niveles de estado C, donde cada uno de los niveles podría apagar determinados relojes, reducir ciertas tensiones suministradas a componentes ociosos, apagar la memoria cache, etc. Mientras más elevado sea el número del estado C, menor será la potencia

demandada pero mayor será la latencia requerida para regresar al estado C0 (estado de ejecución). Algunos procesadores permiten la elección de un estado C por núcleo.

Como ambos estados, C y P, tienen incidencia en la potencia y el tiempo, es necesario evaluar el impacto de ellos en el consumo energético durante operaciones de C/R

Software de Aplicación: Básicamente, un checkpoint consiste en guardar el estado de una aplicación, tal que, en caso de fallo, se pueda reiniciar la ejecución desde ese punto guardado. Cuanto mayor sea el tamaño del problema de la aplicación, mayor será el tiempo requerido para guardar su estado. Su incidencia, al menos en el tiempo, convierte al tamaño de problema en un factor que afecta el consumo energético de operaciones de C/R.

Software de Sistema: Hay dos tipos de software de sistema altamente involucrados en las operaciones de C/R. Por un lado, el sistema que se ocupa de realizar esas operaciones. Por otro lado, como el archivo de checkpoint requiere ser resguardado en almacenamiento estable y remoto, es necesario utilizar un sistema de archivos distribuído. En nuestro caso, el software de sistema que utilizamos es Distributed MultiThreaded CheckPointing (DMTCP)[3] y Network File System (NFS). Ambos softwares tienen opciones de configuración que inciden sobre el tiempo de ejecución y/o la potencia, y por lo tanto son factores que afectan el consumo energético de las operaciones de C/R.

En el caso de NFS, se permite montar carpetas de forma síncrona (opción *sync*) o asíncrona (opción *async*). Si una carpeta de NFS se monta con la opción *sync*, las escrituras en dicho punto de montaje causarán que los datos sean descargados completamente en el servidor NFS, y escritos en almacenamiento persistente, antes de retornar el control al cliente[1]. Así, el tiempo de ejecución de una operación de escritura se ve afectado al variar esta configuración.

En el caso de DMTCP, es una herramienta que realiza checkpoints de forma transparente sobre un grupo de procesos diseminados entre muchos nodos y conectados por sockets, tal como ocurre con aplicaciones MPI (*Message Passing Interface*). DMTCP es capaz de comprimir (utilizando el programa gzip) el estado de un proceso para requerir un menor espacio de almacenamiento en disco y reducir la cantidad de datos transmitidos por la red (entre el nodo de cómputo y el nodo de almacenamiento). El uso o no de la compresión impacta en el tiempo de ejecución y en la potencia requerida (al realizar un trabajo diferente), por lo tanto es otro factor que incide en el consumo energético.

4. Plataforma y diseño experimental

Plataforma experimental

Los experimentos se realizaron sobre un cluster de computadoras con una red Ethernet de 1 Gbps. Cada nodo, tanto de cómputo como de almacenamiento, posee 4 GB de memoria principal, un disco rígido SATA de 500 GB y 7200 rpm,

[1] https://linux.die.net/man/5/nfs

y un procesador Intel Core i5-750, con un rango de frecuencia de 1,2 GHz a 2,66 GHz (sin utilizar el mecanismo Intel Turbo Boost[2]), cuatro núcleos (sin multithreading), y 8 MiB de memoria caché. Los nodos utilizan el sistema operativo GNU/Linux Debian 8.2 Jessie (kernel version 3.16 de 64 bits), OpenMPI version 1.10.1 (implementación de MPI), y la herramienta de checkpoint DMTCP [3] versión 2.4.2. El sistema de archivos de red utilizado para hacer la escritura remota de los archivos de checkpoint es NFS v4.

Las mediciones de potencia se realizaron con el osciloscopio PicoScope 2203 (cuya presición es del 3 %), la sonda diferencial activa TA041, y la pinza de corriente PP264 60 A AC/DC, todos productos de Pico Technology. Las mediciones de tiempo de ejecución del checkpoint y del restart se realizaron utilizando la opción provista por DMTCP.

La aplicación seleccionada para la caracterización del sistema es una aplicación de transferencia de calor del tipo SPMD escrita en MPI que utiliza el tipo de dato float. Esta aplicación describe, mediante una ecuación, el cambio de temperatura en el tiempo, sobre un plano, dada una distribución de temperaturas inicial y ciertas condiciones de borde.

Diseño experimental

Se utilizan dos nodos de cómputo (salvo especificado lo contrario), y cada nodo de cómputo escribe en un nodo de almacenamiento diferente a través de un NFS configurado en modo asíncrono (salvo especificado lo contrario). La tasa de muestreo utilizada para ambos canales del osciloscopio se estableció en 1000 Hz. Las mediciones de potencia corresponden a la potencia disipada por el nodo completo incluyendo la fuente. El DMTCP está configurado para comprimir los archivos de checkpoint (salvo especificado lo contrario). Las pruebas se realizaron con la opción *c-states* del procesador activa (salvo especificado lo contrario). Para cambiar la frecuencia del procesador se utiliza el *governor userspace* de GNU/Linux, el cual permite modificar el archivo sysfs *scaling_setspeed* disponible para cada núcleo. Al modificar la frecuencia siempre se indica la misma para todos los núcleos.

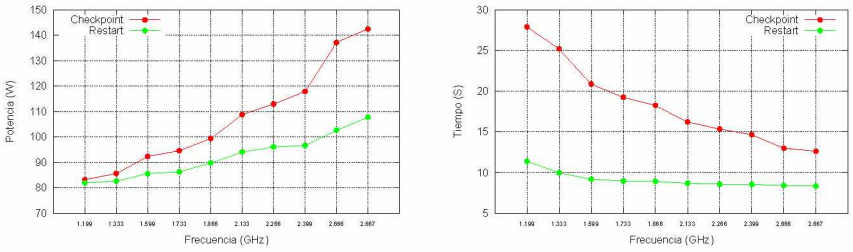
Cada experimento consiste en lanzar la aplicación con un proceso por núcleo, dejarla ejecutar durante un período de precalentamiento de 20 segundos, realizar un checkpoint manualmente, abortar la aplicación (desde el coordinador del DMTCP), y luego volver a reinicarla desde la línea de comandos con el script de restart generado por DMTCP. El tiempo precalentamiento se definió empíricamente observando cuándo la curva de potencia disipada llega a los valores máximos para esa aplicación. Cada experimento se repite tres veces (suficiente para la poca variabilidad de los instrumentos de medición utilizados) y se obtiene la media.

[2] https://www.intel.com/content/www/us/en/architecture-and-technology/turbo-boost/turbo-boost-technology.html

5. Experimentos y Análisis de Resultados

5.1. Estados P del procesador

Se evaluó el impacto de los estados P del procesador en el consumo energético de las operaciones de C/R. La figura 1 muestra la potencia disipada y el tiempo de ejecución para diferentes frecuencias del procesador. Se observa que a medida que aumenta la frecuencia de reloj, la potencia disipada aumenta y el tiempo de ejecución disminuye. Las funciones obtenidas son estrictamente crecientes para el caso de la potencia disipada y decrecientes para el caso del tiempo de ejecución. La diferencia de la potencia disipada y el tiempo de ejecución a la máxima y a la mínima frecuencia llega a ser de casi 60 W y 15 segundos para el checkpoint, y de casi 26 W y 3 segundos para el restart. También se puede observar cómo los cambios en la frecuencia de reloj afectan más a la potencia disipada y al tiempo de ejecución del checkpoint que del restart.



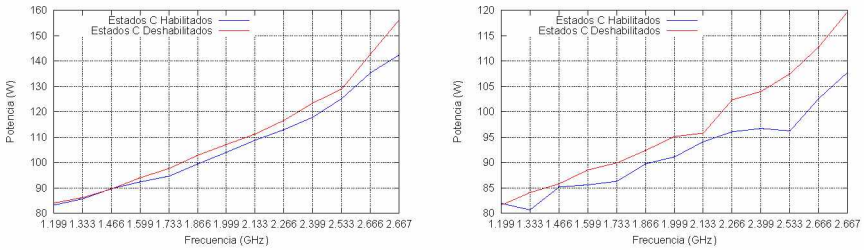
(a) Potencia disipada (b) Tiempo de ejecución

Figura 1: Incidencia de los estados P

5.2. Estados C del procesador

Durante la escritura o lectura de un archivo de checkpoint es posible que el procesador tenga momentos ociosos y por lo tanto sucedan transiciones entre los estados C que afecten la disipación de potencia del procesador. Para estudiar su comportamiento, se realizaron operaciones de C/R con los estados C habilitados y deshabilitados, para todas las frecuencias del procesador.

En la figura 2 se observa que las mediciones de potencia con los estados C habilitados muestran mayor variabilidad, especialmente en el restart. En este gráfico también se observa cómo la diferencia entre la potencia disipada con estados C habilitados y deshabilitados aumenta al aumentar la frecuencia del procesador. Esta diferencia llega a ser aproximadamente del 9 % para el checkpoint (en la frecuencia máxima), y del 10 % para el restart (en la frecuencia 2,533 GHz). Además, impacta en un mayor consumo de energía cuando los estados C están deshabilitados. Para el caso del checkpoint, el consumo llega a ser un 13 % mayor, y para el caso del restart, un 20 % mayor. Los tiempos de ejecución no mostraron variación al habilitar o deshabilitar los estados C.



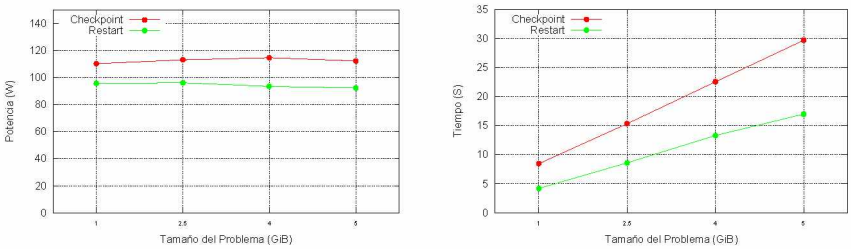
(a) Checkpoint (b) Restart

Figura 2: Incidencia de los estados C en la potencia

5.3. Tamaño del Problema

Se evaluó el impacto del tamaño del problema en el consumo energético de las operaciones de C/R. Se realizaron mediciones de la potencia disipada y el tiempo de ejecución, para diferentes tamaños de problema. No se usaron tamaños que superen la memoria principal disponible en el nodo de cómputo.

En el gráfico 3 se observa que la potencia disipada casi no varía al variar el tamaño del problema (no supera el 4 %), mientras que el tiempo de ejecución, como era de esperar, aumenta a medida que aumenta el tamaño del problema.



(a) Potencia disipada (b) Tiempo de ejecución

Figura 3: Incidencia del tamaño del problema

5.4. Configuración del NFS

Se evaluó el impacto en el consumo energético del uso de la opción *sync* o *async* en el montado del NFS. La figura 4 muestra gráficos que comparan la potencia disipada, el tiempo de ejecución y la energía consumida por un checkpoint almacenado sobre un sistema de archivos de red montado con la opción *sync* y *async*, para tres frecuencias de reloj diferentes (mínima, media y máxima del procesador). Para las tres frecuencias, la potencia disipada es mayor y el tiempo de ejecución es menor cuando se usa la configuración asincrónica. El menor tiempo se explica por el modo de funcionamiento del modo asincrónico, que no necesita esperar que los datos sean descargados en el servidor para

avanzar. La mayor potencia disipada puede deberse a que el modo asincrónico disminuye los tiempos ociosos del procesador. Para la frecuencia mínima y media, las diferencias son pequeñas, dando como resultado un consumo energético similar (un 1.5 % mayor para el caso asincrónico en la mínima frecuencia, y un 7 % menor para el caso asincrónico para la frecuencia media). En ambos casos, al ir mas lento el procesador del nodo de cómputo, no necesita detenerse tanto a esperar la respuesta del servidor NFS. En cambio para la máxima frecuencia esta diferencia se amplía, llegando a un ahorro energético de hasta un 25 % al usar el modo asincrónico.

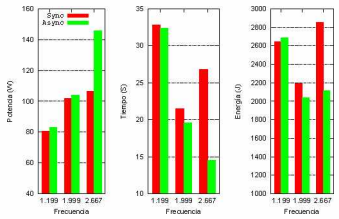
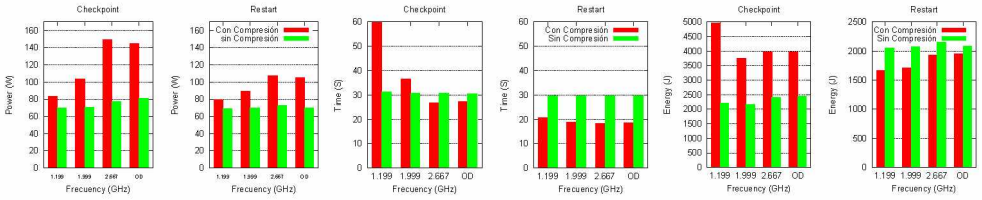


Figura 4: Potencia disipada, tiempo de ejecución y energía consumida por el checkpoint para tres frecuencias.

Debido a su menor consumo energético, y a la menor variabilidad en el tiempo de ejecución, es preferible almacenar los checkpoints en un NFS montado con la opción *async*.

5.5. Compresión de los archivos de checkpoint

Se analizó el impacto de la compresión de los archivos de checkpoint en el consumo energético del checkpoint y del restart. Los experimentos se realizaron sobre un único nodo de cómputo escribiendo en un nodo de almacenamiento.



(a) Potencia Disipada (b) Tiempo de Ejecución (c) Energía

Figura 5: Análisis del impacto de la compresión en el consumo energético.

La figura 5 muestra los gráficos que comparan la potencia disipada, el tiempo de ejecución y la energía consumida por el checkpoint y el restart, con y sin compresión de los archivos de checkpoint, para tres frecuencias de reloj diferentes (mínima, media y máxima disponibles en el procesador) y para el *governor ondemand*.

Los resultados obtenidos muestran lo siguiente:

- La potencia disipada, tanto en el checkpoint como en el restart, es mayor al usar la compresión. Esto se debe al mayor uso de la CPU que se requiere para ejecutar el programa de compresión (gzip) que utiliza DMTCP.
- Sin compresión, el tiempo de ejecución casi no varía para diferentes frecuencia de reloj, tanto para el checkpoit como para el restart.
- El consumo de energía al utilizar el *governor ondemand* es similar al obtenido al utilizar la máxima frecuencia.
- En el checkpoint, el consumo energético siempre es mayor al usar compresión (hasta un 55 % en la frecuencia mínima). Ocurre lo contrario para el restart, donde el consumo energético siempre es menor al usar compresión (hasta un 20 % para el tamaño analizado).
- El menor consumo energético del checkpoint se da para la frecuencia media (1,999 GHz) y sin usar compresión. El menor consumo energético del restartse da para la frecuencia baja (1,199 GHz) y usando compresión.

Teniendo en cuenta que en general se realizan más operaciones de checkpoint que de restart, es conveniente no utilizar compresión para reducir el consumo energético.

6. Conclusiones y trabajos futuros

Este trabajo muestra, a partir de una serie de experimentos sobre un cluster homogéneo ejecutando una aplicación SPMD, cómo impactan diferentes factores del sistema y de las aplicaciones, en el consumo energético del C/R coordinado utilizando DMTCP. Se estudió el impacto de los estados P y C del procesador. Se encontró que las operaciones de checkpoint son más sensibles a cambios en los estados P que las operaciones de restart. Por el contrario, los cambios de los estados C del procesador afectan más al restart. En la plataforma evaluada, el uso de los estados C permite ahorrar energía (hasta 13 % en el checkpoint y 20 % en el restart). El aumento en el tamaño del problema de la aplicación en estudio siempre resulta en un aumento en el consumo energético, debido a su alta incidencia en el tiempo de ejecución de las operaciones. Se encontró que al utilizar la máxima frecuencia de reloj se llega a obtener hasta un 25 % de ahorro energético al utilizar la configuración asíncrona de montaje del NFS. La compresión de los archivos de checkpoint es beneficiosa para el restart, registrando hasta un 20 % de ahorro energético al usar archivos comprimidos, mientras que impacta negativamente en la operación de checkpoint, llegando a tener hasta un 55 % más de gasto de energía al usar la mínima frecuencia de reloj. Entre los trabajos futuros se espera evaluar el impacto en el consumo energético de las operaciones de C/R, del modelo de programación de la aplicación paralela, y de otras herramientas de tolerancia a fallos, así como evaluar la energía consumida por el nodo de almacenamiento.

Referencias

1. Acpi - advanced configuration and power interface. Available at: http://www.acpi.info, accessed on 2018-07-09
2. Amrizal, M.A., Takizawa, H.: Optimizing Energy Consumption on HPC Systems with a Multi-Level Checkpointing Mechanism. Networking, Architecture, and Storage (NAS), 2017 International Conference on. IEEE, 2017 (2017)
3. Ansel, J., Arya, K., Cooperman, G.: DMTCP: Transparent Checkpointing for Cluster Computations and the Desktop pp. 1–12 (2009)
4. Bergman, K., Borkar, et al.: Exascale computing study: Technology challenges in achieving exascale systems. Defense Advanced Research Projects Agency Information Processing Techniques Office (DARPA IPTO), Tech. Rep 15 (2008)
5. Cui, X., Znati, T., Melhem, R.: Adaptive and power-aware resilience for extreme-scale computing. In: 2016 Intl IEEE Conferences on Ubiquitous Intelligence. vol. 00, pp. 671–679 (July 2016), doi.ieeecomputersociety.org/10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0111
6. Dauwe, Daniel, e.a.: Optimizing checkpoint intervals for reduced energy use in exascale systems. Eighth International Green and Sustainable Computing Conference (IGSC). IEEE, 2017. (2017)
7. Diouri, Gluck, O., Lefevre, L., Cappello, F.: Ecofit: A framework to estimate energy consumption of fault tolerance protocols for hpc applications. In: 2013 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)(CCGRID)
8. Diouri, M., Gluck, O., Lefevre, L., Cappello, F.: Energy considerations in checkpointing and fault tolerance protocols. In: Dependable Systems and Networks Workshops (DSN-W), 2012 IEEE/IFIP 42nd International Conference on. pp. 1–6
9. Ibtesham, Dewan, e.a.: Coarse-grained energy modeling of rollback/recovery mechanisms. ependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on. IEEE, 2014. (2014)
10. Le Sueur, E., Heiser, G.: Dynamic voltage and frequency scaling: The laws of diminishing returns. In: Proceedings of the 2010 international conference on Power aware computing and systems. pp. 1–8 (2010)
11. Meneses, E., Sarood, O., Kale, L.V.: Assessing energy efficiency of fault tolerance protocols for hpc systems. In: Proceedings of the 2012 IEEE 24th International Symposium on Computer Architecture and High Performance Computing. pp. 35–42. SBAC-PAD '12, IEEE Computer Society, Washington, DC, USA (2012)
12. Mills, et al.: Energy consumption of resilience mechanisms in large scale systems. In: Proceedings of the 2014 22Nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing
13. Mills, B., Grant, R.E., Ferreira, K.B., Riesen, R.: Evaluating energy savings for checkpoint/restart. In: Proceedings of the 1st International Workshop on Energy Efficient Supercomputing. pp. 6:1–6:8. E2SC '13, ACM, New York, NY, USA
14. Rajachandrasekar, R., et al.: Power-check: An energy-efficient checkpointing framework for hpc clusters. 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing pp. 261–270 (2015)
15. Saito, Takafumi, e.a.: Energy-aware I/O optimization for checkpoint and restart on a NAND flash memory system. Proceedings of the 3rd Workshop on Fault-tolerance for HPC at extreme scale. ACM, 2013. (2013)
16. Shalf, J., Dosanjh, S., Morrison, J.: Exascale computing technology challenges. In: International Conference on High Performance Computing for Computational Science. pp. 1–25. Springer (2010)