프로세스 레벨 전력 소비 프로파일러의 비교

강민재 · 노동건

숭실대학교

Comparison of the Process-level Power Consumption Profilers

Min-jae Kang · Dong-kun Noh*

Soongsil University

E-mail : {mjkang, dnoh}@ssu.ac.kr

요 약

최근 에너지 문제가 사회적인 이슈가 되면서 그린컴퓨팅이 주목을 받고 있다. 그린 컴퓨팅을 위한 다각적인 접근 방법 중 하나로, 컴퓨터의 전력 소비 프로파일링에 관한 연구가 활발히 진행되고 있는데, 대표적으로 PowerAPI, PowerTop, JouleMeter, pTop, 그리고 EnergyChecker 등의 도구가 있다. 이러한 연구들은 순수 소프트웨어에 기반하여 각 컴퓨터 디바이스들의 소비 전력을 측정할 수 있을 뿐 아니라, 이를 바탕으로 프로세스 레벨에서 전력 소비 프로파일링을 수행함으로써 각 프로그램의 전력 소비량도 분석할 수 있다. 따라서 전력을 많이 소비하는 프로세스를 파악하고 제어함으로서 전체 전력 소비를 줄이거나, 프로그램 설계 시 프로파일링 전력 데이터를 바탕으로 저전력 프로그램설계를 가능케 하여, 궁극적으로 그린 컴퓨팅을 지향할 수 있게 한다. 본 논문에서는 전술한 대표적인 연구들을 비교 분석하여 이상적인 프로세스 레벨 전력 소비 프로파일러의 특징들을 도출하고자 한다.

ABSTRACT

Recent social issues is energy issues, green computing has attracted attention. Active research on the power consumption of computer profiling is one of the various approaches for green computing. As a representative tool PowerAPI, PowerTop, JouleMeter, pTop, and EnergyChecker. These studies can be used to measure the power consumption of each computer device because it is based on a pure software. Based on this profiling process at the level of power consumption by performing the power consumption of each program can be analyzed. Therefore to identify the processes that consume a lot of power and control the total power consumption by reducing. also when designing the program, based on data profiling power enables the design of low-power programs, and ultimately can be oriented green computing.

In this paper, by comparing and analyzing the associated representative studies, the ideal process level will draw on the characteristics of the power consumption profiler.

키워드

리눅스, 전력 소비, 에너지 소비, 프로파일러, 프로세스 레벨, 에너지 측정 도구

Ⅰ. 서 론

유비쿼터스 환경이 가속화됨에 따라 컴퓨터와 모바일 디바이스의 에너지 사용량은 지속적으로 증가 하고 있고, 이에 따라 2020년에는 탄소배출 비율이 15%까지 증가 할 것으로 예측[1]되고 있다. 따라서 에너지 자원 절약 및 친환경 에너지원의 사용을 모토로 하는 그린컴퓨팅이 최근 화두가 되고 있다.

컴퓨팅 디바이스의 에너지 사용량을 줄이기 위하여, 우선 하드웨어 또는 소프트웨어 단위의 에너지 소비 패턴에 대한 이해가 선행되어야 한다

이 주제에 대해서 최근 많은 연구들이 이루어지고 있는데 대표적으로 pTop, PowerAPI, AppScope, JouleMeter, PowerTOP, EnergyChecker, 그리고 Quanto 등이 있다.

본 논문에서는 전술한 대표적인 연구들을 비교 분석하여 이상적인 프로세스/디바이스 레벨 전력 소비 프로파일러의 특징을 도출하고자 한다.

본 논문의 2장에서는 pTop에 대하여 설명하고 있고, 3장과 4장에서는 PowerAPI와 AppScope, 그리고 5장에서는 그 외의 연구들에 대하여 기술한다. 마지막으로 6장에서는 각 연구들을 비교하고, 결론에서 최적의 프로파일러의 조건을 도출하고 있다.

Ⅱ. pTop

pTop[2]은 리눅스 플랫폼을 타겟으로 만들어진 에너지 미터링 도구로써 최근에는 윈도우 플랫폼에서도 동작하도록 포팅되었다. 응용프로그램을 위한 파워 제어 및 모니터링 라이브러리(API)를 제공하는 특징을 갖는다.

2.1 에너지 모델

시간 t동안 어플리케이션 E_{appi}의 에너지 사용량은 아래와 같이 계산될 수 있다.

$$E_{appi} = \sum U_{ij} \times E_{resourcej} + E_{interaction}$$

U_{ij}는 자원 j에서의 어플리케이션 i의 사용량이고, $E_{resourcej}$는 resource j에 의한 에너지 소비량이고, $E_{interaction}$은 시스템 자원 사이의 상호작용에 의한 간접적인 에너지 소비량이다.

특정 자원의 에너지 소비는 해당 자원의 상태(read, write, etc)와 전송으로 결정 된다. 시간 t동안 상태 S와 전송 T의 에너지 사용량은 아래와 같이 계산될 수 있다.

$$E_{resourcej} = \sum_{j \in S} P_j t_j + \sum_{k \in T} n_k E_k$$

이 수식은 다른 자원들에게도 보편적이고 쉽게 적용될 수 있다.

캐싱, 버퍼링과 같은 시스템 정책에 의해 결정되어지는 $E_{interaction}$에 대해서는 향후 연구 과제로 남겨두었다.

2.2 하드웨어 정보

프로세스와 시스템에 대한 정보는 procfs와 sysfs에서 얻어 온다. 예를 들어 CPU에서 에너지 계산은 "/sys/devices/system/cpu/"에서 얻어온 정보를 기반으로 계산한다.

에너지 소비에 대한 정보는 이상적으로 OS(Operating System)와 하드웨어 벤더가 각 상태에서 하드웨어의 에너지 소비량을 제공 하여야 한다. 하지만 이와 관련하여 자세한 스펙이 제공되지 않는 경우도 있는데, 이러한 경우에는 에너지 소비량을 사용자가 따로 설정할 수 있다.

2.3 오버헤드 & 정확도 측정

pTop의 오버헤드와 정확도 측정을 위해 노트북에서 60개 이상의 프로세스를 실행시켜 테스트를 하였다. top 명령어를 사용하여 pTop의 오버헤드를 측정한 결과 CPU는 3%의 사용률을 보였고, 메모리는 전체 메모리의 0.15%를 사용하였다.

정확도는 Watts Up이라는 하드웨어 기반 에너지 측정 장비를 사용하여 측정 하였다. 측정 결과는 그림 1에서 볼 수 있듯이 5% 이내의 오차를 보였다.

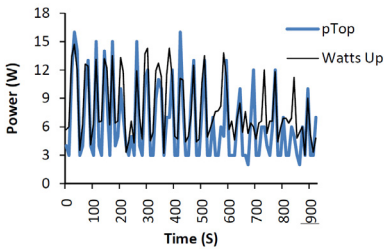

그림 1. pTop 정확도 측정

Ⅲ. PowerAPI

PowerAPI[1][3][4][5]는 리눅스 플랫폼을 대상으로 구현된 도구이며 pTop과 마찬가지로 응용프로그램을 위한 에너지 관련 API를 제공한다.

3.1 에너지 모델

소프트웨어 c의 에너지 사용량은 아래와 같이 계산될 수 있다.

$$E_c = E_{comp} + E_{com} + E_{infra}$$

E_{comp}는 컴퓨터자원의 비용(예를 들어 CPU processing, Memory access, I/O operations)이고, E_{com}은 네트워크에서 데이터 교환 비용, 그리고 E_{infra}는 추가적으로 발생한 비용(예를 들어 Java VM)이다.

3.2 하드웨어 정보

프로세스와 시스템에 대한 정보는 pTop과 같은 procfs와 sysfs에서 얻어 온다. 에너지 소비에 대한 정보는 Thermal Design Power(TDP) 공식을 이용하여 계산한 후에 데이터베이스화 시킨다

PowerAPI에서 사용한 TDP공식은 아직까지는 특정 프로세서 그룹에서만 유효하고 다른 프로세서 아키텍처에 대해서는 향후 연구 과제로 남겨 두었다.

3.3 오버헤드 & 정확도 측정

PowerAPI의 정확도 측정은 Powermeter라는 하드웨어 기반 에너지 측정 장비를 사용하여 측정 하였다. 측정 결과는 그림 2에서 볼 수 있다.

오버헤드의 측정은 하노이 타워 프로그램(10개의 타워)을 50회 실행시켜 평균 실행 시간의 차이를 통해 측정 하였다. 하노이 타워만 실행하였을 때 실행시간은 22.64ms, 프로파일러 데몬 작동 중 실행하였을 때 실행 시간은 22.74ms로 평균 오버헤드는 1.79%로 측정 되었다.

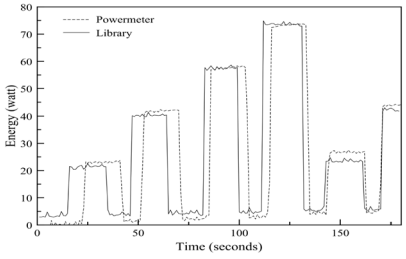

그림 2. PowerAPI 정확도 측정

Ⅳ. AppScope[6][7]

최근 스마트폰 보급이 많아지면서, 안드로이드에 관한 연구도 많이 이루어지고 있다. AppScope는 많은 안드로이드 플랫폼에서의 에너지 소비 프로파일링 연구 중 대표적인 것중 하나이다.

4.1 에너지 모델

에너지 모델은 보통 선형(linear)과 비선형(non-linear)으로 구분된다. 비선형 모델이 선형 모델보다 현저히 뛰어나지 않음에도 불구하고 비선형 모델은 하드웨어 컴포넌트 간에 의존적 이다. 그래서 AppScope에서는 선형 모델만을 고려하였다.

디바이스의 에너지 소비 P는 아래와 같다.

$$P = \sum_i (\beta_i \times x_i) + P_{base} + P_{\in}$$

i는 컴포넌트, x_i는 사용량의 벡터, β_i는 계수, P_{base}는 기본적인 에너지 소비량, $P_{\in}$은 모델에서 측정할 수 없는 에너지 소비량이다. 스마트폰의 전체 에너지 소비량 E는 아래와 같다.

$$E = \sum_j E^j + (P_{base} + P_{\in}) \cdot D, \text{ where}$$
$$E^j = \sum_i (\beta_i \times x_i^j) \cdot d_i^j$$

D는 디바이스의 작동 기간, E^j는 프로세스 j에 의한 에너지 소비량 이다.

4.2 하드웨어 정보

리눅스의 경우 프로세스와 시스템에 대한 정보를 procfs와 sysfs에서 얻어왔다. 하지만 안드로이드의 경우 여러 가지 문제(permission, etc)로 인해 procfs와 sysfs에 접근하는 것이 쉽지 않다. AppScope에서는 프로세스와 시스템에 대한 정보를 얻기 위해 안드로이드 플랫폼에서 제공하는 BatteryStats이라는 API를 사용하였는데, 이 API는 procfs와 sysfs의 정보를 그대로 계승하고 있다.

4.3 오버헤드 & 정확도 측정

AppScope의 정확도 측정은 Monsoon이라는 하드웨어 기반 에너지 측정 장비를 사용하여 측정하였다. 그림 3이 정확도 결과를 보여주고 있는데 50초~60초에 Monsson에서 측정한 값이 갑자기 증가한 것을 볼 수 있다. 이 증가값은 Slave 4에서 3G사용을 위해 Wifi를 끄는데, 이 때 3G의 사용이 시스템에 의해 자동적으로 실행되기 때문에 발생한다. 그림에서 알 수 있듯이 AppScope의 전체 에너지 측정값은 Monsson과 5.2%의 차이가 있다.

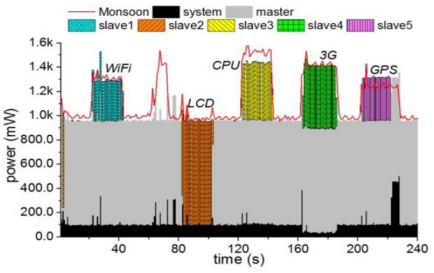

그림 3. AppScope 정확도 측정

AppScope의 오버헤드는 그림 4에서 보이는 것처럼 표준편차 1.9와 최악의 경우 5.9%의 오버헤드를 보인다.

Ⅴ. 파워 미터링 도구들의 비교

앞 절에서 소개한 연구들 이외의 대표적인 프로파일러는 PowerTOP[8], JouleMeter[9], EnergyChecker[10], Quanto[11] 등이 있다.

PowerTOP의 경우 리눅스 플랫폼에서 동작하며, library를 제공하지 않는다. JouleMeter는 윈

표 1. 연구 비교

	DVFS	이용률(%)	정확도(%)	플랫폼	Library	추가장비	시스템정보
pTop	O	3(cpu)/0.15(memory)	95	Linux Windows	O	X	procfs, sysfs
PowerAPI	O	1.75	N/A	Linux	O	X	procfs, sysfs
AppScope	O	5.9	94.8	Android	N/A	X	batterystats
PowerTop	N/A	N/A	N/A	Linux	X	X	N/A
JouleMeter	O	N/A	N/A	Windows	X	X	N/A
EnergyChecker	N/A	N/A	N/A	Linux	N/A	O	N/A
Quanto	O	N/A	N/A	TinyOS	N/A	X	N/A

도우즈 플랫폼에서 동작하며 Microsoft에서 만들었고, 내부 소스가 공개되지 않은 블랙박스 형태로 제공된다. EnergyChecker는 하드웨어 기반 에너지 측정 장비가 추가로 필요하며, Quanto는 TinyOS 플랫폼에서 동작한다.

앞서 소개한 도구들에 대한 특성을 표 1에서 요약/비교하고 있다. DVFS(Dynamic Voltage and Frequency Scaling)를 사용하는 시스템인 경우 Thermal Design Power(TDP)를 이용하여 에너지 소모를 예측하는 경우가 많았는데, TDP를 이용할 경우 비교적 쉽게 에너지 사용량을 계산할 수 있으나, 정확한 결과를 도출하지는 못한다는 단점도 있다. pTop의 경우 TDP를 이용하지 않고, 사용자가 직접 각 디바이스별 전력 사용량을 입력해 유효한 전력 소비 값을 계산할 수 있었다. 반면 PowerAPI의 경우 TDP를 이용해 유요한 전력 사용량을 계산할 수 있었지만 특정 프로세서 아키텍처에서만 유효하게 적용될 수 있었다.

몇 가지 도구들은 에너지 모니터링 및 제어 관련 Library는 API형태로 제공하여 일반 앱개발자들도 쉽게 사용할 수 있는 환경을 제공하였다.

VI. 결론

앞 절에서 분석한 각 연구들의 특성을 종합하여 이상적인 프로파일러의 특징을 도출해보면 아래와 같다.

- 추가적인 하드웨어가 필요하지 않아야 할 것
- 추가적인 입력 없이 유효한 전력 사용량을 TDP 이외의 다른 방법으로 모델링 하여 정확도를 높일 수 있어야 할 것
- On/Offline 결합 등으로 시스템 오버헤드를 최소화하는 방법으로 구현되어야 할 것
- 재사용 가능한 Library를 지원할 것
- 다양한 플랫폼의 지원이 가능할 것

참고문헌

[1] A. Bourdon, A. Noureddine, R. Rouvoy, and L. Seinturier, "Linux: Understanding Process-Level Power Consumption,"2nd International Workshop on Green Computing Middleware(GCM'11), Lisbon, Portugal, December 2011

[2] Thanh Do, Suhib Rawshdeh and Weisong Shi, "pTop: A Process-level Power Profiling Tool," in Proceedings of the 2nd Workshop on Power Aware Computing and Systems (HotPower'09), Big Sky, MT, USA, October 10, 2009

[3] A. Noureddine, A. Bourdon, R. Rouvoy, and L. Seinturier, "e-Surgeon: Diagnosing Energy Leaks of Application Servers," Inria Research Report, January 2012

[4] A. Noureddine, A. Bourdon, R. Rouvoy, and L. Seinturier, "A Preliminary Study of the Impact of Software Engineering on GreenIT," In 1st International Workshop on Green and Sustainable Software (GREENS'12/ICSE'12), Zurich, Switzerland, June, 2012

[5] A. Noureddine, A. Bourdon, R. Rouvoy, and L. Seinturier, "Runtime Monitoring of Software Energy Hotspots," In 27th International Conference on Automated Software Engineering (ASE'12), Essen, Germany, September 2012

[6] C. Yoon, D. Kim, W. Jung, C. Kang, H. Cha, "AppScope: Application Energy Metering Framework for Android Smartphone using Kernel Activity Monitoring," 2012 USENIX Annual Technical Conference (USENIX ATC'12), Boston, MA, USA, June 2012

[7] W. Jung, C. Kang, C. Yoon, D. Kim and H. Cha, "DevScope: A Nonintrusive and Online Power Analysis for Smartphone Hardware Components," 2012 International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS'12), Tampere, Finland, October 2012

[8] Intel Open Source Technology Center, "https://01.org/powertop/"

[9] "Joulemeter USER MANUAL," Microsoft Research, June 2011

[10] aaron-tersteeg, Intel Software, "http://software.intel.com/en-us/articles/intel-energy-checker-sdk/"

[11] Rodrigo Fonseca , Prabal Dutta , Philip Levis , Ion Stoica, "Quanto: Tracking Energy in Networked Embedded Systems," Proceedings of the 8th USENIX conference on Operating systems design and implementation(OSDI'08), Berkeley, CA, USA, 2008