Webpage-Based Benchmarks for Mobile Device Design

Marc Somers and JoAnn M. Paul
Virginia Tech
Electrical and Computer Engineering
Blacksburg, VA 24061
{msomers106, joann.paul}@gmail.com

Abstract

Computers are currently designed using benchmarks and specification styles that are decades old, even as computers are being used in fundamentally different ways. By investigating the content, structure and usage of webpages, we observe that webpages represent a fundamentally different standard for performance evaluation of computers. We gathered data and modeled typical webpage content in order to characterize what is becoming a uniquely important design space. We then included this data in a set of simulations that also included models of a variety of scheduler types and heterogeneous multiprocessor architectures. To this, we proposed usage patterns that we believe typify the way people access the Internet on mobile devices. Considering only modern-day content in webpages, we found that specialized architectures can improve performance up to 70% over a homogeneous multiprocessor composed of general purpose processors with 25% additional improvement over the next best architecture when individual user preferences are also considered. This trend will increase as webpages become more differentiated in purpose and more complex in content. A new model of performance evaluation of computing must be developed, based upon webpage content and webpage access patterns.

1. Introduction

Virtually all computation is becoming more mobile; whether considering cell phones that access the internet, or laptops that can be used for voice communications, the computing landscape is changing, even as the importance of mobile computing is becoming apparent. And yet, computers are still designed using benchmarks and specification styles that are decades old, even as computers are being used in fundamentally different ways.

As cell phones take on more data processing capability, to include communications with the Internet, they are starting to take on features of a more general class of computing. The result is a completely unique class of computing for which no adequate performance models exist. Embedded system benchmarks contain sets of applications that are intended to execute one at a time, while a typical webpage represents a collection of jobs, of varying size and degree. Increasingly, webpages are becoming the defacto standard of information exchange and users are accessing webpages on mobile devices. Without accurate benchmark models, designers have no hope of finding the best design for mobile devices that access the Internet.

The Apple iPhone captures a trend in mobile computing, boasting functionality of a phone, web browser, camera, movie player, and music player. And yet, mobile computers are still designed with a mindset used for embedded systems – that the behavior of the system is periodic and specified to a fixed set of tasks. Performance evaluation in general purpose computing does not pose an answer either, as it relies upon programmable benchmarks where applications are evaluated one at a time. Computer design requires a new model against which performance is evaluated.

In this paper, we do not propose to develop an entirely new benchmark suite. Rather we illustrate, through experimentation, that webpage content and usage patterns already differ sufficiently in content to point to different designs. Webpage content and access patterns represent a form of benchmarks that is currently not utilized. Our contribution is the observation that webpage form and content has matured enough for consideration as the foundation of a benchmark suite.

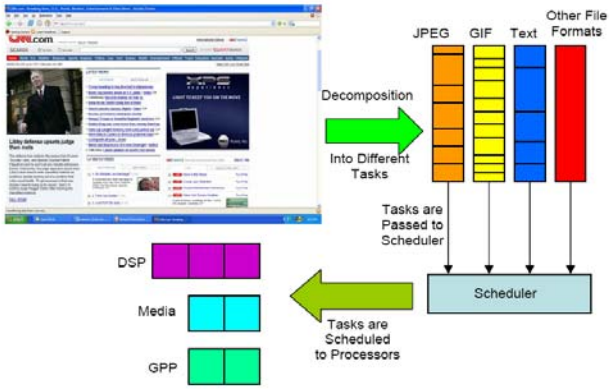

Figure 1. Webpage as Performance Benchmarks

2. Background

Mobile device users have incorporated the Internet into their daily lives [1], which means that future mobile device designs need to consider accessing websites and executing their respective contents as a means of determining architecture performance. Figure 1 illustrates how each webpage provides a diverse set of heterogeneous applications that other benchmarks only provide individually. When a webpage is accessed, the tasks are decomposed into smaller task types such as JPEG, FLASH, and GIF file formats. These task types are taken by a scheduler and distributed, in some fashion, to the different processors associated with the device's architecture.

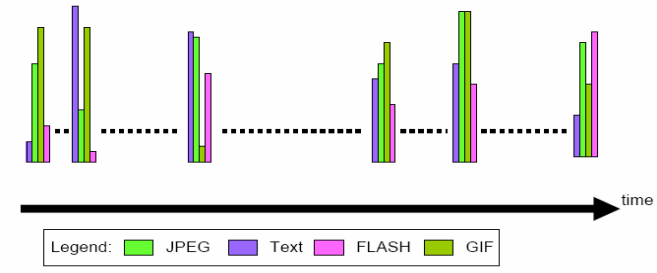

Figure 2. Bursty nature of webpage access

978-1-4244-1922-7/08/$25.00 ©2008 IEEE

The type of scheduling strategy employed in the design affects performance and must be taken into consideration at design time.

Figure 2 displays the input from the heterogeneous webpage applications over time. It shows a collection of jobs of different types that arrives simultaneously as the user accesses a webpage. The vertical bars in the figure reflect a collection of jobs of different sizes – different working set size or different complexity – but of the same type. Webpage content is highly variable in size and quantity. In addition, large bursty jobs come into the device which must be processed simultaneously. At the same time, Single Chip Heterogeneous Multiprocessors (SCHMs) make the design space even more complex opening up possibilities for entirely new design strategies, but only if the means of evaluation matches what users actually do with computers.

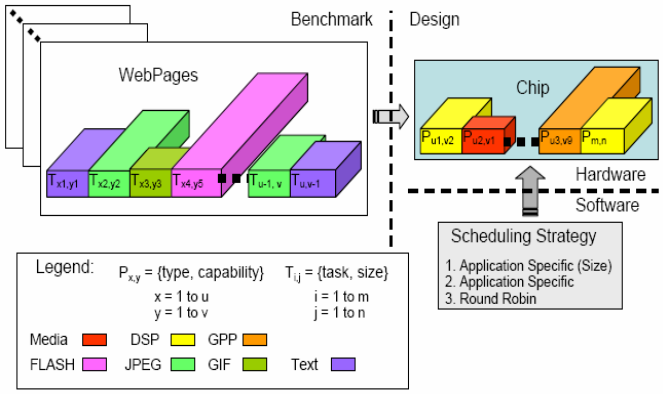

Figure 3. Webpage interaction with chip architecture

3. Model Description

Figure 3 presents an abstraction of our model parameters. There are three key elements of the model portrayed in Figure 3: webpage content, scheduling and processor models. The figure shows that mobile devices have three categories which can be manipulated at high levels for significantly different overall performance. These are hardware, scheduling and applications.

We assume that the numbers and types of processors that exchange information via shared memory represent the most significant architectural features. Other hardware issues, such as communications and memory organization, only add to the size of the design space, which we believe will only strengthen our argument. However, they are beyond the scope of our experimentation. Thus, we show processors of different types in each bar on the chip, and the numbers of each type are projected into the chip on the drawing. We selected a number of schedulers, to include custom schedulers, which we discuss later in our experimentation. The schedulers are presumed to take a collection of jobs and determine when and where they should be scheduled. The applications are the webpages.

Like the design undergoing evaluation, the webpages contain different types, again represented as bars, and different numbers of jobs of the same type. Webpage content has an additional degree of complexity associated with the size of individual jobs. What is significant about Figure 3 is that the content of the webpages is similar in form to the evaluation space of the chip, when the chip is viewed as a heterogeneous multiprocessor.

Note that individual tasks need not be parallelized in the processing of a webpage – tasks come in parallel form, based solely on webpage content.

3.1 Webpage Content

A number of existing benchmark suites exists, all focused on the evaluation of a single program at a time, whether for general purpose programming, embedded systems, or even the evaluation of a single application executed concurrently on a parallel computer. The SPEC CPU [2] is the most widely used benchmark by computer architects [3]. Examples of "embedded system" benchmark suites include Mibench [3], MediaBench [4], and EDN Embedded Microprocessor Benchmark [5]. MediaBench focuses on complete applications for multimedia and communications systems [4, 6]. By contrast to benchmark-based evaluation, high-level modeling and simulation languages such as System-C and SpecC [7] are really specification languages for desired system behavior. As such, these are based upon an assumption that the underling system specification is periodic, which is consistent with the foundations of the high-level design languages (HDLs) such as Verilog and VHDL from which they were derived. Thus, neither traditional benchmarks nor specification languages accurately capture the behavior portrayed in Figures 2 and 3.

Most webpages are composed of three basic elements: text/scripts, images, and movie/animated FLASH applications. A webpage also consists of links to many objects that need to be downloaded and processed in order to view a webpage correctly. Figure 4 displays a screenshot of the MLB webpage, which was modified to point out the different elements of the webpage.

Figure 4. Screenshot of MLB webpage

Consider the news websites for BBC and CNN. Despite providing similar information, these websites differ in the content that each webpage provides. Both of these websites; however, are more text-based oriented than the other websites being analyzed.

Two sporting news websites are ESPN and MLB. ESPN is a general sporting news website while MLB is more specialized for a particular sport and organization. The content associated with each webpage varies on a daily basis, more so for ESPN depending on the day of week and the sporting events that are occurring. ESPN tends to provide more headline stories on its main page while MLB provides links to stories. Both have

approximately the same number of image files but ESPN has more FLASH files, although they are smaller sized FLASH files.

The last category of website we considered is one which we will call EDU, but it is drawn from www.vt.edu. This webpage is geared towards capturing attention and providing college information. This site is more geared towards media/image content and provides a unique alternative of webpage content to be analyzed versus the news and sporting news webpages.

There are three types of movie/animated FLASH applications. The first type is MPEG type movies that the website provides a link to so users can download the packets and view the movie file. The second type is a FLASH movie very similar to the MPEG type movie with frames and audio, but instead it is built with FLASH animation and is typically executed when a user visits a website. The last type is a still image FLASH frame. These still images tend to be in a rotation pattern beginning with a still image and then proceeding to the next image of a series of images after short delays. The still image FLASH files are similar to regular images in processing. Instead of multiple frames and audio, these files tend to have a singular frame with no audio simplifying the task of processing these types of files. MPEG and animated FLASH files take hundreds of times longer to process than all other files on a website collectively. [9]

Table 1. Statistical Data of Websites (compiled using [10]).

Statistics	BBC	CNN	ESPN	MLB	EDU
Total Objects	214	414	92	75	124
Total Size (KB)	269.0	438.1	343.5	116.4	394.1
Total JPEG	10	8	10	5	24
Total GIF	193	390	61	57	86
Total FLASH	0	0	10	3	1
Total TEXT	11	16	11	10	13
Media Content	94.9%	96.1%	88.0%	86.7%	89.5%

Table 1 shows a summary of job types and sizes by webpage category that we collected for our model of webpages. The majority of webpage content tends to be media oriented as in image or movie/animation files, a trend which will continue.

Table 2. Webpage Categorization

File Types/Sizes	BBC	CNN	ESPN	MLB	EDU
JPEG > 50KB	0	0	0	0	2
JPEG > 10KB	0	1	2	0	2
JPEG < 10KB	10	7	8	5	20
GIF > 50KB	0	0	0	0	0
GIF > 10KB	0	1	1	1	0
GIF < 10KB	200	380	60	56	16
FLASH > 50KB	0	0	1	2	1
FLASH > 10KB	0	0	4	1	0
FLASH < 10KB	0	0	5	0	0
Text > 50KB	1	0	0	0	0
Text > 10KB	2	6	6	1	2
Text < 10KB	8	10	5	9	11

Table 2 depicts an approximation of the various file types with the webpages. The elements in the webpages provide insight into categorizing the webpages to aid in the webpage utilization portion of the experiments, discussed later. Notice the differences between the webpages that provide similar services.

Based upon the job types that dominate virtually all webpages, it seems reasonable to select three categories of processor types to simulate: a Media Processor, a Digital Signal Processor (DSP), and a General-Purpose Processor (GPP).

3.2 Processors

GPPs have the largest variety of individual chip makers and types ranging from a couple GHz to low 100 MHz clock speeds. DSPs are second in terms of choice selection followed by the Media processor. The number of processors being considered was determined by the benchmarked data available and tested results of the different processors on various applications and tasks. The most useful benchmarking site related to individual task types was the EDN Embedded Microprocessor Benchmark Consortium (EEMBC) [5]. Still, each benchmarking organization uses their own in-house scoring scheme and not all processors can be compared against all attributes.

The processors we used in our experiments are referred to as Media, DSP, and GPP processors throughout the paper where Philips PNX1700 is the Media processor, Analog Devices ADSP Blackfin533 is the DSP, and the AMD K6-2E+ is the GPP. All processors were geared towards an embedded system, thus making these exact processors inadequate for use in a mobile device such as a cell phone due to size and power requirements. Detailed information that would allow comparison of the range of processor types used in actual cell phones is currently proprietary, and our goal was to cover a heterogeneous design space. However, relative performance of these processors can be related to the relative performance of processors used in mobile devices. Table 3 has the relative performance for each task on every processor while Table 4 has the overall relative performance encompassing all the tasks; these results were obtained from experiments performed and published by EEMBC. Note that higher numbers mean faster execution.

Table 3. Relative Performance for Each Task

Task Type	ADSP-BF533	PNX1700	AMD K6-2E+
JPEG	1	8.93	1.81
Text	2.936	1	5.015
MPEG	1.28	4.383	1

Table 4. Overall Relative Performance

Task Type	ADSP-BF533	PNX1700	AMD K6-2E+
JPEG	14.294	127.642	25.868
Text	17.536	5.973	29.952
MPEG	1.28	4.383	1

The Media processor is the best for images and movies, especially when those media files need to be processed in large quantities. Although consuming more power, the GPP is the second best choice for image files, interesting since the DSP barely outperforms the GPP for movie files. Both the DSP and GPP are significantly faster for text processing than the Media processor, the GPP is the fastest. The processors are compared in Table 5.

Table 5. Relative Area and Power Consumption Comparison

Processor	Relative Area	Relative Power
AMD-K6E (500 MHz)	3.3998	19.8571
PNX1700 (500 MHz)	1.0000	4.0286
ADSP-BF533 (594 MHz)	1.5772	1.0000

The GPP processor, AMD-K6E+, is the worst processor in terms of size and power consumption. The GPP processor is three times larger than the media processor and two times larger than the DSP processor. In terms of power consumption, the GPP consumes 5

times more power than the media processor and almost 20 times the power of a DSP processor. Since the GPP processor is the best processor with respect to text processing by being twice as fast as the DSP processor and five times as fast as the media processor, the GPP processor is needed to achieve the best system runtime performance. However, the quantity of GPP processors available in the system and the tasks to be scheduled for the processor should be limited in order to consume less energy. The Tables also suggest that the media processor is essential for a mobile device due to its small size and excellent performance for image and movie files. The only disadvantage for the media processor is its inability to deal well with text processing. However, supplementing a media processor with DSP and GPP processors could prove invaluable in creating an architecture that achieves great performance while minimizing area and power.

Table 6. Architectural Comparisons

Architecture	Power (W)	Rel. Power	Rel. Size	Area ($mm^{2)}$	Rel. Area
2 GPP	27.8	7.94	6.80	4957	1.04
5 DSP	3.5	1	7.89	5749	1.20
8 Media	22.6	6.45	8	5832	1.22
2 GPP, 1 Media	30.6	8.75	7.80	5686	1.19
1 GPP, 2 DSP	15.3	4.37	6.55	4778	1
1 GPP, 2 DSP, 1 Media	18.1	5.18	7.55	5507	1.15
1 GPP, 1 DSP, 3 Media	23.1	6.59	7.98	5815	1.22
1 GPP, 4 Media	25.2	7.19	7.40	5394	1.13
4 DSP, 1 Media	5.6	1.61	7.31	5328	1.12
3 DSP, 3 Media	10.6	3.02	7.73	5636	1.18
2 DSP, 4 Media	12.7	3.62	7.15	5216	1.09
1 DSP, 6 Media	17.6	5.03	7.58	5524	1.16

3.3 Architectures

Table 6 displays the comparative values of the different architectures we tested. The architectures were created based on an equivalent size of the Video iPod architecture, which encompasses approximately eight relative size units. Thus, we normalized our processors to what we believe would be 1/8 of an iPod for one size unit. Also shown are relative and absolute power values. Thus we investigated a total of 3 homogeneous architectures, 7 two processor-type architectures, and 2 three processor-type architectures. We presumed shared memory in all cases – further experimentation could differentiate on the basis of communications as well.

3.4 Schedulers

A total of four scheduling strategies were employed. The schedulers were round robin (RR), static (SS), application specific scheduler that knows job sizes (ASJ), and application specific scheduler that does not know job sizes. For the application specific scheduler that does not know job sizes, the order that threads (representing jobs) were created influenced the jobs placement in the job queue. The application specific scheduler where threads were created from biggest to smallest was called 'Application Specific Big' (ASB) while the scheduler with threads created from smallest to biggest was called 'Application Specific Normal' (ASN).

3.5 Overhead

The inclusion of different processors for consideration by the same scheduler requires additional cost/overhead. This cost is associated with having separately compiled copies of the same task available for the different processor types. It is not unreasonable to assume that architecture with three different processor types uses approximately three times the memory than a homogeneous architecture with the same number of processors. Other work indicates that the cost of storage for more complex scheduling strategies can be ignored [11]. Most authors that have covered scheduler overhead agreed that the scheduler's computational overhead is insignificant versus the computational demands of the applications. However, we model the performance overhead associated with the application specific schedulers as estimated from [12]. An extrapolation of the linear relationship was performed to predict the overhead due to the number of tasks associated with the various webpages used for the experiments. A constant value was used for the application specific schedulers overhead even though the application specific scheduler's overhead should decrease as the number of available tasks decreases. The constant value was used to provide an upper bound on the overhead's possible impact when scheduling tasks. Table 7 shows the overhead associated with the schedulers.

Table 7. Scheduler Performance Overhead

Scheduler	RR	ASJ	ASB	ASN	SS
Cycles	159	4770	1590	1590	159
Time (secs)	5.06E-05	1.52E-03	5.06E-04	5.06E-04	5.06E-05

3.6 Simulation Environment

In this work we use MESH (Modeling Environment for Software and Hardware) because it permits performance and power evaluation when threads execute on sets of heterogeneous resources under a variety of custom scheduling [13]. Power is calculated as the per-cycle wattage times the number of cycles the processor executes, averaged over time. This is done at a high level. Thus, MESH is capable of simultaneously modeling and simulating all of the design elements of Figure 3 at a high level, and also evaluating the timed nature of job arrival as in Figure 2.

4. Experimental Results

Our experiments are designed to illustrate how webpages are important models of classes of applications and application structure that should be developed into benchmark suites. Importantly, we do not develop a benchmark suite in this paper – such a suite would investigate over all webpages in order to completely characterize the domain of webpages. However, we will show that sufficient differentiation in current webpages exists, so that such a suite should be developed.

In other work [14] we consider other applications that may be carried out simultaneously by cell phones – the focus of this work is on wepbages because we have observed the way webpages are becoming the defacto standard of information exchange. We do not claim to have developed webpage benchmark suites, nor to have converged on final designs. We do intend to motivate the importance of getting the structure of the design space correct – in both benchmark and design parameters – so that designers can evaluate for what computers actually do now and in the future.

Our experimental results are broken down into two parts. In the first part, we average performance over all webpage types. In the second part, we consider how a user accesses webpages of particular types. We normalized all performance results against the performance of a homogeneous multiprocessor, using GPPs. Thus, unit performance is for a homogeneous multiprocessor, and smaller umbers mean faster time to process the entire content of a webpage – lower numbers mean faster performance.

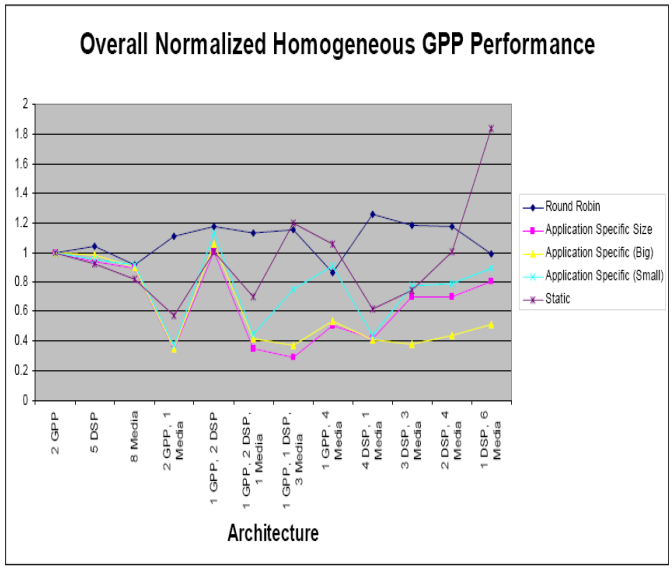

Figure 5. Overall Normalized GPP Performance

4.1 Average Performance Over All Webpages

Figure 5 shows that the best performing designs have heterogeneous architectures and application specific schedulers. When using complex scheduling strategies, the architectures with multiple media processors where the total media processors were less than half the total processors saw the greatest performance gains – ranging from 20% to 60%. The performance gains attributed to the usage of the application specific scheduling strategies reside in the exploitation of the heterogeneous nature for both architecture and applications.

4.2 Customization to Webpage Utilization

Considering the wide variation in the kinds of information on different webpage types and the likelihood that individual users will access the webpages they prefer far more frequently than ones they do not prefer, we next evaluated the performance of the different architectures and scheduling strategies over anticipated webpage utilization. Table 8 defines various user profiles and their respective webpage usage frequency. Our profiles were developed intuitively, based upon profile name. While we do not claim to have done profiling experimentation, a separation of webpage access based upon individual user interest in the content of webpages seems a reasonable assumption. We also included energy modeling in this set of experiments, since the MESH tool is capable of power modeling for heterogeneous multiprocessors.

This experimentation is typical of many kinds of design in which adding more variables to the design space dramatically increases the overall numbers of designs that can be considered. Accordingly, we developed multiple "cuts" to narrow the selection of architectures we investigated. While we do not claim our methodology for narrowing our design selections to be comprehensive, we note the importance of developing a more robust cut methodology as we expand the parameters in the evaluation space. Note that even the need to develop such a new design methodology is another indication of the importance of developing an overall strategy for webpage-based design.

The first 'cut' was derived by taking the average for the performance, energy consumption, and performance-energy product individually and comparing to the homogeneous GPP. The second 'cut' session focused on eliminating the number of similar architectural styles, such as the three processor-type architectural designs where over half the total processors were media processors. The final 'cut' session focused on reducing redundancies between the GPP/Media and DSP/Media architectures with respect to the number of media processors.

Table 8. Webpage Utilization (in %) for Various User Profiles

Type of Person	BBC	CNN	ESPN	MLB	EDU
International Political Junkie	90	10			
Political Junkie	75	20	5		
Web Surfer	20	20	20	20	20
Political & College Sports Enthusiast	25	20	20		35
Sports Fanatic	0	0	75	15	10
Typical College Student	65		25		10

Figure 6 depicts performance for different webpage utilizations, again normalized to a homogeneous multiprocessor with GPPs.

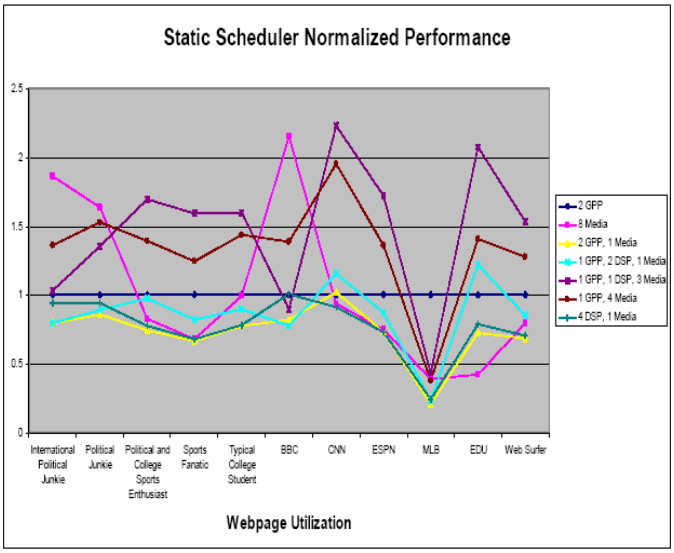

Figure 6. Performance for Webpage Utilization

Figure 6 shows that different utilization patterns favor different architectures. While the "2 GPP, 1 Media" architecture might seem to be the best overall performer, it performs 20% worse than the next two best performers for pure CNN users and almost 50% worse than the best architecture for pure EDU users.

Even considering only modern-day content in webpages, specialized architectures can improve performance up to 70% over a homogeneous multiprocessor composed of general purpose processors with 25% additional improvement over the next best architecture when individual user profiles are also considered.

Figure 7 shows the performance-energy product for the static scheduler when the device is constantly in operation, i.e. accessing a webpage frequently enough so that it is never worth incurring the penalty of restart.

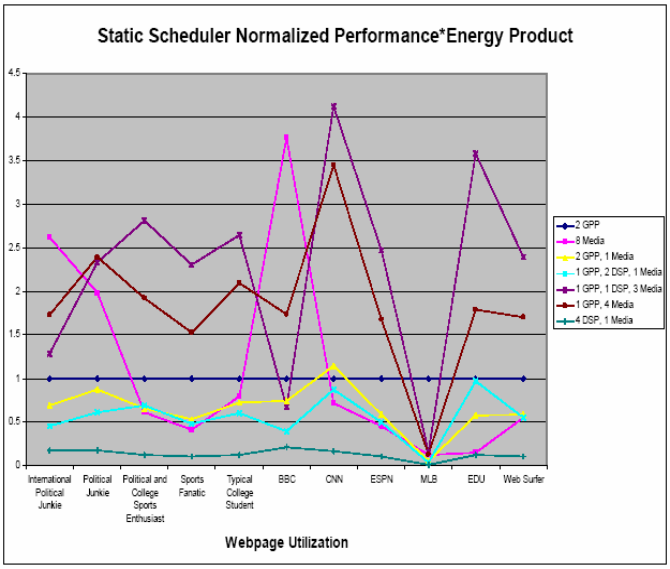

Figure 7. Performance*Energy for Webpage Utilization

The "4 DSP, 1 Media," is the best performer. However, that same architecture is no better in terms of performance than the baseline case of pure GPPs for pure BBC users, as shown in Figure 6, and is almost the same as the baseline case for the user profile which we dubbed, "Political Junkies." If a news oriented person did not have their device on all of the time, unlike a sports enthusiast who follows multiple games, they would do better with a different architecture.

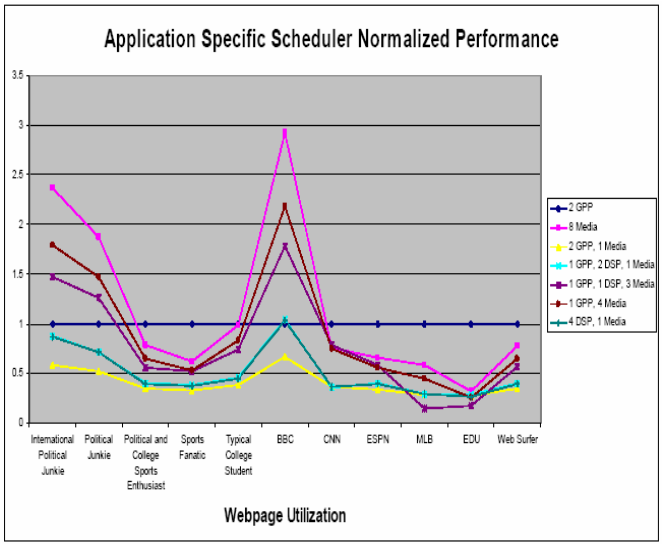

Figure 8. ASN Performance Over Webpage Utilization

Figure 8 explores the same architectures when using the ASN scheduler. While the AS schedulers tend to favor, overall, the architecture with "2 GPP and 1 Media processor," the curves intersect the closer a user gets to one who largely visits the MLB or EDU websites. While scheduling techniques have impact, there is still benefit to customizing devices to webpage access profiles.

5. Conclusions

The Internet is becoming the standard by which humans communicate – for timely news information and business, as well as recreation. And yet, designers still evaluate computer systems for applications that execute one at a time, not in the structure of webpages, which is that of a highly variable collection of job types and sizes. We showed that performance differentiation of designs based upon webpage content and access patterns exists. The differential can be expected to grow as webpage content becomes more significant, diverse and complex. It can also become more significant as displays and other human computer interface mechanisms change the nature of the way webpages are accessed. We showed that a new webpage-based benchmark suite is required, which will fundamentally alter computer design. This work is believed to be the first to point in such a direction.

6. Acknowledgements

This work was supported in part by the National Science Foundation under grants 0607934 and 0606675. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

7. References

[1] M. Meeker, B. Pitz, B. Fitzgerald, R. Ji. "Internet Trends," www.morganstanley.com/institutional/techresearch/pdfs/Internet_Trends1005.pdf. October 2005.

[2] http://www.spec.org/benchmarks.html.

[3] M. Guthaus, J. Ringenberg, D. Ernst, T. Austin, T. Mudge, R. Brown. "MiBench: A free, commercially representative embedded benchmark suite," Proceedings 4th Workshop on Workload Characterization, pp. 1-12, 2001.

[4] C. Lee, M. Potkonjak, W. Mangione-Smith. "MediaBench: A Tool for Evaluating and Synthesizing Multimedia and Communications Systems," ISCA '97.

[5] http://www.eembc.org/.

[6] B. Bishop, T. Kelliher, M. Irwin. "A Detailed Analysis of Mediabench," Workshop on Signal Processing Systems, '99

[7] R. Domer, D. Gajski, A. Gerstlauer. "SpecC Methodology for High-Level Modeling," 9th IEEE EDP Workshop, 2002.

[8] http://neuron2.net/LVG/ratesandsizes.html.

[9] http://www.half-serious.com/swf/format/

[10] http://www.websiteoptimization.com/services/analyze/

[11] Y. Cho, S, Yoo, K. Choi, N. Zergainoh, A. Jerraya. "Scheduler Implementation in MP SoC Design," DAC '05.

[12] W. Yuan, K. Nahrstedt. "Energy-Efficient Soft Real-Time CPU Scheduling for Mobile Multimedia Systems," 19th Symp. on Operating System Principles, 2003, pp 149-163.

[13] B. Meyer, J. Pieper, J. Paul, J. Nelson, S. Pieper, A. Rowe, "Power-Performance Simulation and Design Strategies for Single-Chip Heterogeneous Multiprocessors," IEEE Transactions on Computers, vol. 54, Jun. 2005, pp. 684-697.

[14] J. Paul, D. Thomas, A. Bobrek, "Scenario-Oriented Design for Single-Chip Heterogeneous Multiprocessors," *IEEE Transactions on VLSI*, Vol. 14, Aug. 2006, pp. 868-880.