GPPRMon: GPU Runtime Memory Performance and Power Monitoring Tool

GPPRMon

Burak Topçu, Işıl Öz

Department of Computer Engineering, Izmir Institute of Technology

August, 2023

Introduction

- Requirement for **high-performant** and **energy-efficient** heterogeneous computer systems.
  - GPUs play a crucial role as accelerators or co-processors.
- Addressing overall energy consumption and runtime power issues for GPU-based systems more [1].

Table: Quanitative exploitation metrics among accelerators and co-processors [2].

	Accelerator/Co-Processor	Count	System Share (%)
1	NVIDIA Tesla V100	61	12.2
2	NVIDIA A100	27	5.4
3	NVIDIA A100 SXM4 40 GB	18	3.6
4	NVIDIA Tesla A100 80G	10	2
5	NVIDIA Tesla V100 SXM2	10	2
6	AMD Instinct MI250X	10	2
7	NVIDIA Tesla A100 40G	9	1.8
8	NVIDIA A100 SXM4 80 GB	7	1.4
9	NVIDIA H100	5	1
10	NVIDIA Tesla P100	5	1

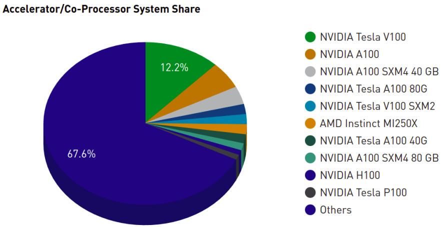


Figure: Accelerator and co-processor system share distribution [2]

Introduction

- Catastrophic performance and energy efficiency problems, especially for the execution of memory-intensive workloads.
  - Memory wall problem still exists [3, 4].
  - Performance- and energy-improving solutions compete with each other.
  - **More analytical observations** are necessary for design decisions [5, 6].
- Complicated GPUs due to recent developments such as tensor cores, concurrent kernel execution, and SER.
  - More detailed research with empirical data to point out current bottlenecks for increasing the performance and throughput.

Outline

- Executive summary
  - Problems, motivation, and contributions.
- Background
  - GPU programming environment.
  - GPU architectures and the state-of-the-art GPU simulator.
- *GPPRMon* methodological overview.
  - Collected micro-architectural performance and power metrics.
  - Metric collection and visualization options.
  - Visualizations:
    - General View
    - Temporal View
    - Spatial View
- Descriptive case study
- Conclusion

Executive Summary, Problems

- None of NVIDIA's GPU profilers or simulators directly report runtime GPU performance and power consumption.
  - Evaluating the GPU application performance and energy consumption at kernel basis hides most of the detailed observations.
  - Contemporary approaches to investigate GPU execution behavior at lower executable granularity are insufficient.
- Several in-house target-specific repetitive works for monitoring the runtime performance and power consumption cause redundant effort in literature.

Executive Summary, Motivation

- Providing a tool to monitor GPU execution temporally and spatially, and track power dissipation at runtime.
  - Supporting all official NVIDIA GPUs.
- Empirically highlighting performance bottlenecks with various execution granularity.
- Saving researchers from repetitive in-house efforts applied to identify performance bottlenecks of various research fields.

Executive Summary, Contributions

- GPPRMon which is built upon GPGPU-Sim [7],
  - is a runtime performance and power monitoring tool for GPU executions.
  - has multiple configuration options depending on the researcher's expectations.
  - visualizes the execution with the general view, temporal view, and spatial view options.
  - is beneficial to identify overall and temporal performance bottlenecks of GPU executions by pointing out the activation of the smallest execution elements.
- A case study describing how to use and utilize GPPRMon.

Programming Environment for NVIDIA GPUs (CUDA, PTX, SASS)

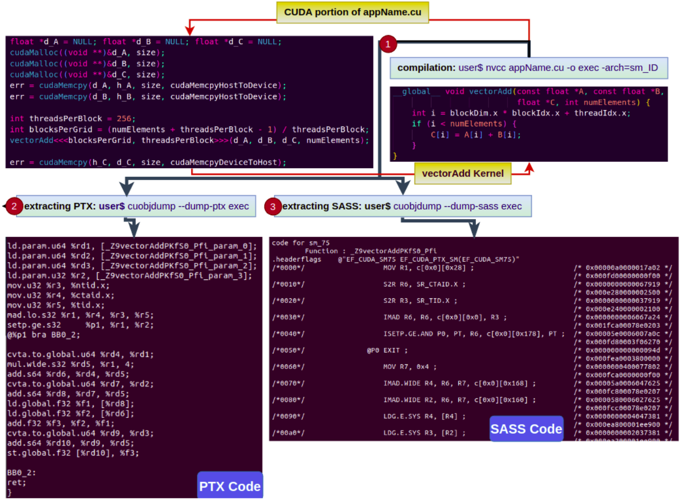


Figure: The programming environment for GPU programs at different levels.

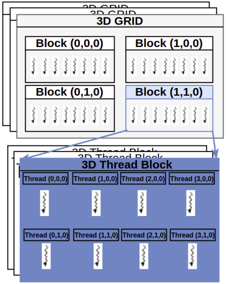


Figure: A kernel structure describing a grid and a block.

NVIDIA GPU Architectures

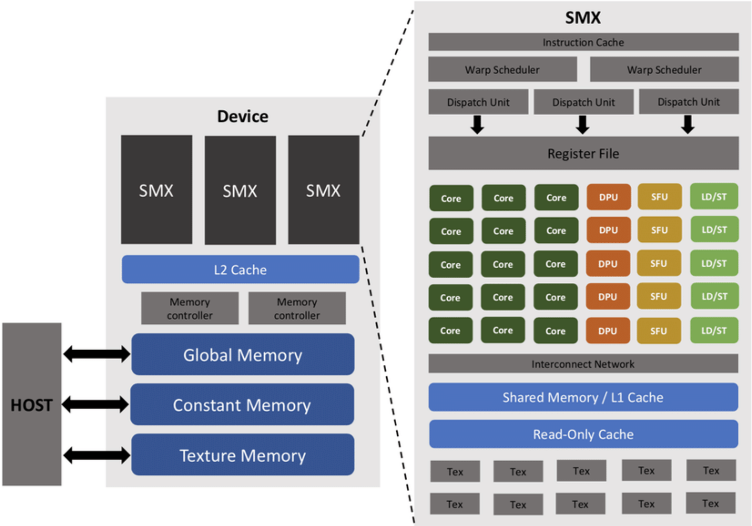


Figure: Typical GPU architecture overview.

Simulator Workflow

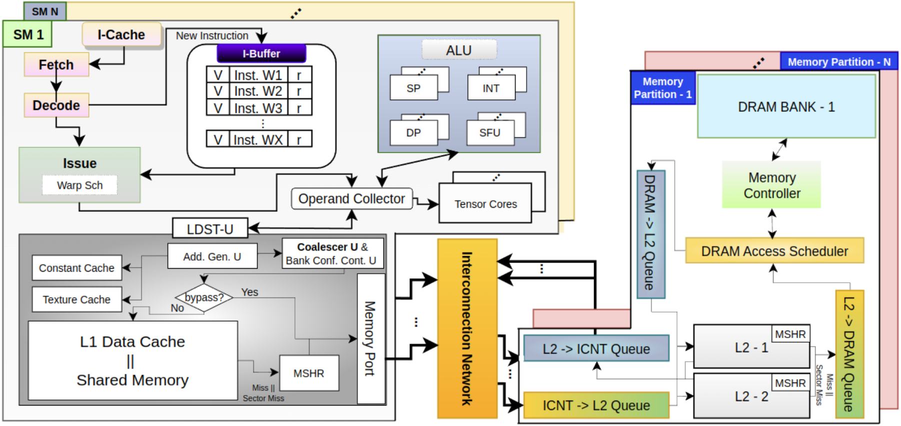


Figure: GPGPU-Sim simulator workflow diagram.

- The GPPRMon is built upon the cycle-accurate performance [7] and the power [8] models of GPGPU-Sim v4.2.

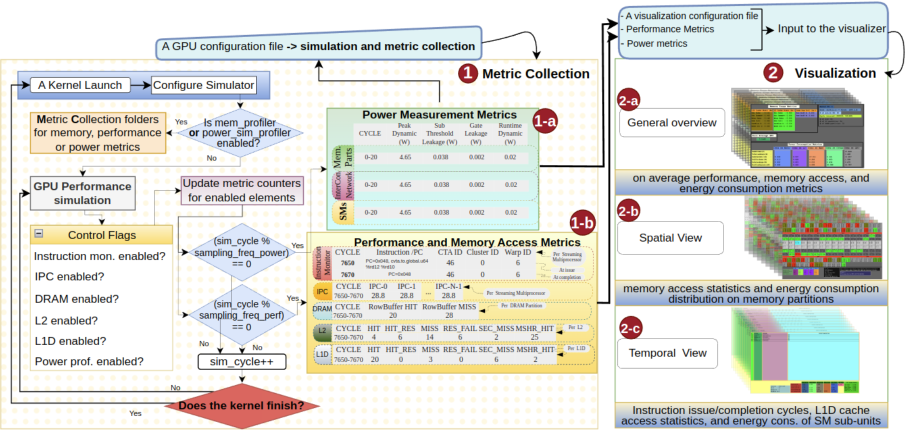


Figure: GPPRMon [a] methodological overview.

[a] https://github.com/parsiyte/GPPRMon

Collected Micro-architectural Performance and Power Metrics

Performance metrics:

- On each L1 data and L2 caches.
  - Hit, hit reserved, miss, sector miss, reservation failure, and MSHR Hit.
- On each row buffer of DRAM banks.
  - Hit and miss.
- Warp instruction issue/completion trackings.
- IPC per SM.

Power Dissipation Metrics:

- Reached *peak dynamic* power dissipation.
- Observed *sub-threshold leakage* and *gate leakage* dissipation.
- *Runtime dynamic* power dissipation.

Metric Collection and Visualization Configuration Options

Table: Configuration options for collecting micro-architectural performance metrics.

Memory profiler	Performance metric collection enable
Memory sampling freq.	Determining the sampling freq. for metric collection
IPC	IPC rate collection enable
Instruction Monitor	Enabling instruction issue/completion recording
L1D Metrics	L1D cache access statistic collection enable
L2 Metrics	L2 cache access statistic collection enable
DRAM RB Metrics	Row buffer statistics collection enable
Store Enable	Including store instructions among statistics
Accumulate Stats	Accumulating statistics throughout the execution

Table: Configuration options for visualization.

Plot GPU	Generates general overview visuals
Plot Memory Hierarch	Generates spatial view visuals
Plot Core	Generates temporal view visuals
Sampling Frequency	Determines per sample interval of execution
GPU Name	GPU Architecture name (GV100, RTX2060 etc.)
Simulation Output	Simulation output (in performance mode) file name
CTA IDs	Thread block IDs to be tracked (0,2,4... or all)

Table: Configuration options for collecting power dissipation.

Power simulation mode	To enable collecting power consumption metrics
Runtime sampling freq.	The sampling freq. for power metric collection
DVFS Enabling	Turning on/off DVFS for the power model
Aggregate Stats	Aggregate power consumption statistics

- GPPRMon's general view displaying memory access statistics, kernel specs, performance, and dissipated power at runtime.

On Average Memory Access Statistics

L1D Cache Stats (Av)	
Hit Rate	0.003
Hit Reserved R	0.001
Miss Rate	0.038
Reserv. Failure R	0.944
Sector Miss R	0.013
MSHR Hit R	0.008

L2 Cache Stats (Av)	
Hit Rate	0.312
Hit Reserved R	0.000
Miss Rate	0.464
Reserv. Failure R	0.000
Sector Miss R	0.223
MSHR Hit R	0.005

DRAM Row Util. (Av)	
Row Buffer H	0.383
Row Buffer M	0.617

Kernel ID: 0

Cycle Interval: [55000, 56000]

Grid:(1784,1,1) Block:(256,1,1)

# of active SMs: 80

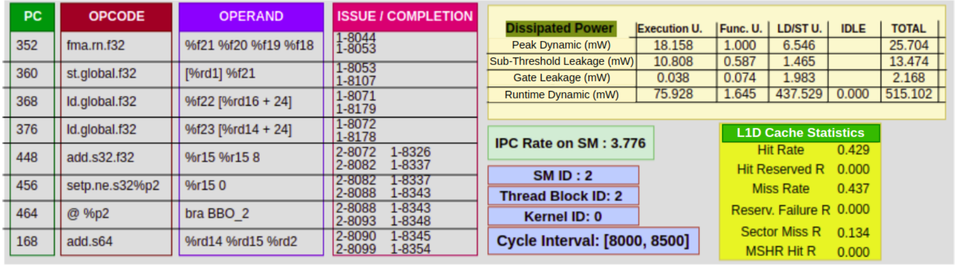
Average IPC on SMs : 1.08

Dissipated Power	InterCon. Net	L2	Mem Part.	SMs	GPU
Peak Power (mW)					185.63
Total Leakage (mW)					17.346
Peak Dynamic (mW)	0.338	4.687	137.55	25.704	168.264
Sub-Threshold Leak (mW)	0.067	0.138	1.316	13.474	14.995
Gate Leakage (mW)	0.011	0.013	0.016	2.168	2.352
Runtime Dynamic (mW)	64.618	2.537	823.184	205.174	1095.513

Figure: GPPRMon general view.

GPPRMon - Temporal View Visualization

- GPPRMon’s temporal view monitoring thread block’s execution, SM performance, and power distribution on SM components at runtime.



PC	OPCODE	OPERAND	ISSUE / COMPLETION
352	fma.rn.f32	%f21 %f20 %f19 %f18	1-8044 1-8053
360	st.global.f32	[%rd1] %f21	1-8053 1-8107
368	ld.global.f32	%f22 [%rd16 + 24]	1-8071 1-8179
376	ld.global.f32	%f23 [%rd14 + 24]	1-8072 1-8178
448	add.s32.f32	%r15 %r15 8	2-8072 1-8326 2-8082 1-8337
456	setp.ne.s32%p2	%r15 0	2-8082 1-8337 2-8088 1-8343
464	@ %p2	bra BBO_2	2-8088 1-8343 2-8093 1-8348
168	add.s64	%rd14 %rd15 %rd2	2-8090 1-8345 2-8099 1-8354

Dissipated Power	Execution U.	Func. U.	LD/ST U.	IDLE	TOTAL
Peak Dynamic (mW)	18.158	1.000	6.546		25.704
Sub-Threshold Leakage (mW)	10.808	0.587	1.465		13.474
Gate Leakage (mW)	0.038	0.074	1.983		2.168
Runtime Dynamic (mW)	75.928	1.645	437.529	0.000	515.102

IPC Rate on SM : 3.776

SM ID : 2

Thread Block ID: 2

Kernel ID: 0

Cycle Interval: [8000, 8500]

L1D Cache Statistics	
Hit Rate	0.429
Hit Reserved R	0.000
Miss Rate	0.437
Reserv. Failure R	0.000
Sector Miss R	0.134
MSHR Hit R	0.000

Figure: GPPRMon temporal view.

- GPPRMon's spatial view reveals memory units' efficiency at runtime.

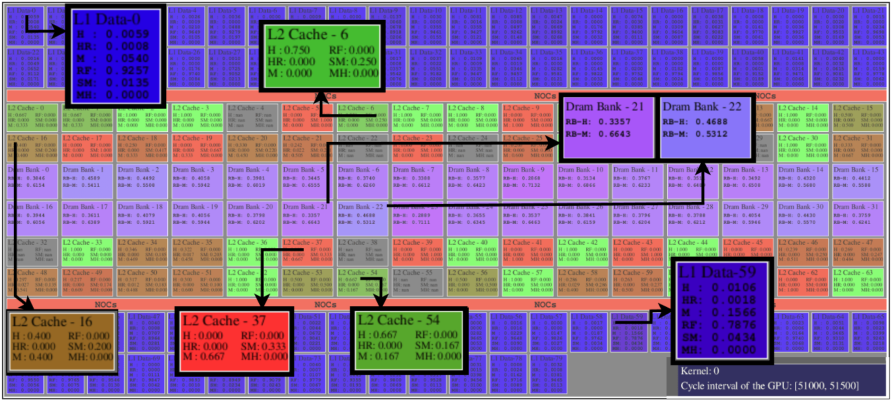


Figure: GPPRMon spatial view.

A Case Study for Performance Bottleneck Analysis with GPPRMon

Table: GV100 GPU specs.

80 SMs with Specifications	Register bank size, # of register bank	65536 32-bit registers, 16 register banks
	SP, SF, DP, INT, TC, LD/ST (WriteBack-PipeDepth)	4, 4, 4, 4, 4, 1(8)
	Warp Scheduler	4 (LRR) per SM
	on-chip L1D Cache, #of banks, access latency, cache line	128KB (4 sets, 64-way), 4, 20 cycles, 128B
	on-chip L1I Cache, #of banks, access latency, cache line	128KB (64 sets, 16-way), 1, 20 cycles, 128B
32 Memory Partitions with Specifications	L2 Cache, #of banks, access latency, cache line	96KB (32 sets and 24-way), 2, 160 cycles, 128B
	DRAM, NoF banks, access latency (after L2)	1GB, 16 banks, 100 cycles, 128B
	DRAM scheduler	First-ready, first-come first-service

Table: Naive performance overview of PR algorithm with web-Stanford on GV100.

Kernel	GPU IPC	GPU Occupancy	L1D		L2		DRAM	Tot. Cycle
			Miss Rate	Res. Fail Rate	Miss Rate	Res. Fail Rate	RowBuf. Loc. (LD+ST)	
Page Ranking - Contrib K0	715.59	82.76%	1.000	0.819	0.333	0.0	-nan	8670
Page Ranking - PullStep K1	3.007	5.55%	0.584	0.400	0.156	0.011	0.658	8677889
Page Ranking - LinNorm K2	1297.68	77.108%	0.501	0.285	0.457	0.001	0.724	11718

- CUDA implementation of Page Ranking (PR) Algorithm [9].
  - Assigning weights to graph nodes depending on relative importance among them.
  - Processing with web-Stanford data [10].
  - Execution with a memory-intensive workload
- We experiment on Volta architecture-based GV100 server GPU.

GPPRMon, Performance Bottleneck Analysis

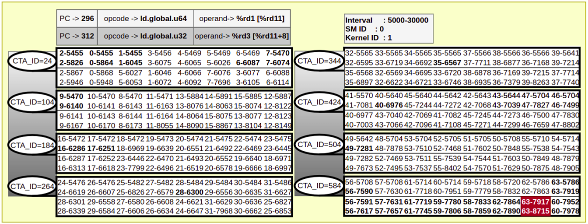


Figure: Temporal View of issue/completions for memory requests belongs to SM0.

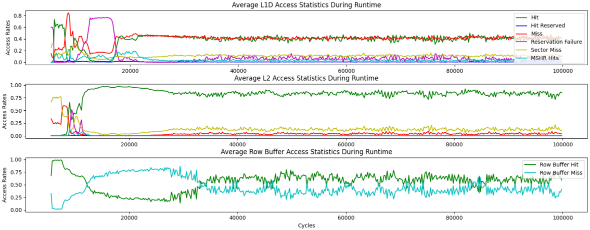


Figure: Runtime memory access statistics on L1D/L2 caches and DRAM row buffers.

- Issue/completion times of warps belonging to 8 thread blocks on SM0 for 2 load instructions.
  - L1D access: 20/25 cycles
  - L2 access: 180/190 cycles
  - DRAM access: 280/290 cycles
- Loads get slower 10 times compared to the ideal performance.

Key 1: The efficiency of off-chip memory workload handling has a crucial impact on overall performance.

GPPRMon, Performance Bottleneck Analysis

Figure: A portion of a Spatial View displaying memory utilization.

- Pressure on L1D cache causes misses, cache pollutions, and early evictions.
- Increased off-chip memory workload mostly hits on L2 caches after 9th snapshots.
  - Data size and data sparsity.

Key 2: Memory utilization behavior changes **temporally** (on L1D caches part 1,2,3) and **spatially** (on L2 caches part 7).

GPPRMon, Impacts of Bottlenecks to the Power

Table: Runtime power consumption distribution among GPU components (in milliwatts).

Cycles	Streaming Multiprocessor: Execution Units	Funct. Units	LD/ST. Unit	SM Idle	**SM Total**
5000, 5500	2637.57	54.30	35.67	23.75	2751.30
5500, 6000	597.03	6.31	860.01	0	1463.69
6000, 6500	614.408	12.41	399.89	0	1026.71
6500, 7000	708.63	14.38	464.29	0	1187.312
7000, 7500	686.78	13.90	463.94	0	1164.62
7500, 8000	795.81	16.26	487.37	0	1299.44
8000, 8500	543.02	10.19	335	0	888.21
8500, 9000	354.47	5.58	249.28	0	609.34
9000, 9500	474.91	5.34	455.42	0	935.67
9500, 10000	446.76	4.76	475.46	0	926.99

Cycles	Memory Partition: MC FEE	PHY Int.	MC Trans. E.	DRAM	L2	NoCs	**MP + NoCs**
5000, 5500	3.74	8.17	4.59	0	0	0.67	16.51
5500, 6000	177.24	17.77	9.39	557.15	3.36	26.92	764.92
6000, 6500	56.06	31.28	16.14	1346.38	3.06	92.60	1452.94
6500, 7000	65.52	31.40	16.20	1354.31	3.05	94.17	1470.52
7000, 7500	65.57	31.37	16.19	1354.91	3.07	94.39	1471.15
7500, 8000	69.97	29.68	15.34	1264.33	3.70	96.66	1383.07
8000, 8500	60.43	30.52	15.76	1341.36	4.4	124.73	1451.96
8500, 9000	51.96	31.10	16.05	1362.57	19.04	148.04	1480.74
9000, 9500	80.23	27.75	14.38	1096.85	66.13	216.84	1284.35
9500, 10000	78.17	23.83	12.42	843.20	41.01	153.79	968.65

The main power contributor components are SMs, memory partitions, and interconnection networks.

The overall power is mostly dissipated by the memory hierarchy.

Conclusion

- We propose a runtime performance and power dissipation monitoring tool, GPPRMon.
  - Collecting several micro-architectural metrics
  - Providing multi-functional visualization options that allow analyzing the execution from various perspectives.
- Performance bottleneck analysis of a sparse graph workload on GV100.
  - Describing how to utilize GPPRMon.
- Overcoming additional in-house works conducting analysis to compare baseline and solution implemented versions.
- Expansion options by integrating CUPTI metric collection to GPPRMon visualization.

References I

S. Mittal and J. S. Vetter, “A survey of methods for analyzing and improving gpu energy efficiency,” *ACM Comput. Surv.*, vol. 47, aug 2014.

TOP500, “List statistics among accelerators and co-processors,” June 2023.

P. Jain, A. Jain, A. Nrusimha, A. Gholami, P. Abbeel, K. Keutzer, I. Stoica, and J. E. Gonzalez, “Checkmate: Breaking the memory wall with optimal tensor rematerialization,” *CoRR*, vol. abs/1910.02653, 2019.

J. Hong, S. Cho, and G. Kim, “Overcoming memory capacity wall of gpus with heterogeneous memory stack,” *IEEE Computer Architecture Letters*, vol. 21, no. 2, pp. 61–64, 2022.

Y. Sun, S. Mukherjee, T. Baruah, S. Dong, J. Gutierrez, P. Mohan, and D. Kaeli, “Evaluating performance tradeoffs on the radeon open compute platform,” in *2018 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pp. 209–218, 2018.

A. Krzywaniak, P. Czarnul, and J. Proficz, “Gpu power capping for energy-performance trade-offs in training of deep convolutional neural networks for image recognition,” in *Computational Science – ICCS 2022*, (Cham), pp. 667–681, Springer International Publishing, 2022.

M. Khairy, Z. Shen, T. M. Aamodt, and T. G. Rogers, “Accel-sim: An extensible simulation framework for validated gpu modeling,” in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*, pp. 473–486, 2020.

References II

V. Kandiah, S. Peverelle, M. Khairy, J. Pan, A. Manjunath, T. G. Rogers, T. M. Aamodt, and N. Hardavellas, "Accelwattch: A power modeling framework for modern gpus," in *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*, MICRO '21, (New York, NY, USA), p. 738–753, Association for Computing Machinery, 2021.

Z. Xu, X. Chen, J. Shen, Y. Zhang, C. Chen, and C. Yang, "Gardenia: A graph processing benchmark suite for next-generation accelerators," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 15, pp. 1280–1293, jan 2019.

J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community Structure in Large Networks: Natural Cluster Sizes and the Absence of Large Well-Defined Clusters," *Internet Mathematics*, vol. 6, no. 1, pp. 29 – 123, 2009.