First of all, thank you for writing me a letter of recommendation. I appreciate the time and effort it takes to write a letter. To help you out, below are a few of the things that I believe most define me and my research. Below are links to my draft research statement and teaching statement, etc. I am planning on submitting my applications in the next few days. At that time, I will send you a message with the details of how to submit the letter to each institution. I don't know the exact deadlines for letter submission, but I expect it's within a couple of weeks.

Don't hesitate to let me know if there is anything else you would like to know.

Research statement: http://cs.wisc.edu/~powerjg/files/research_statement.pdf
Teaching statement: http://cs.wisc.edu/~powerjg/files/teaching_statement.pdf
CV: http://cs.wisc.edu/~powerjg/files/cv.pdf (also: http://cs.wisc.edu/~powerjg)
Cover letter: http://cs.wisc.edu/~powerjg/files/cover_letter.pdf
For letter writers (this email): http://cs.wisc.edu/~powerjg/files/for_letter_writers.pdf
Teaching evaluations from Fall '15: http://cs.wisc.edu/~powerjg/files/teaching_evals.pdf (excerpts of feedback: http://cs.wisc.edu/~powerjg/files/teaching_excerpts.pdf)

My research contributions broken up into two main components: my research on bridging the gap between CPUs and GPUs and my research enabling efficient real-time big-data analytic database queries. My Google Scholar page (https://scholar.google.com/citations?user=7m7TYQsAAAAJ&hl=en) shows my works are already impacting the community. I have over 120 citations with 44 to the gem5-gpu paper, 39 to the HSC paper, and 25 to the GPU address translation paper. I've been invited to give industry talks at AMD Research, Google, and IBM.

Beyond these contributions, when given an opportunity, I say "yes". I have taught an undergrad lecture (~150 students), am a prolific contributor to the gem5 community, and have multiple other collaborations in the CS department.

CPUs + GPUs - from physically integrated to logically integrated

- Problem: GPUs are good for many workloads, but much of the time is spent transferring data between CPU and GPU memory. This was even true for integrated GPUs when I started this work. Also, many workloads need more than the 1-8 GB of DRAM on discrete GPUs
    - Logically integrating the GPU into the same memory model as the CPU simplifies programming significantly. With my work, there is now no need to transform pointer-based datastructures or manually copy data from the CPU to GPU memory.
    - Performance can be improved as well. I found that for the scan application I investigated in the database workloads, over 95% of the execution time was wasted doing data copies.
- Heterogeneous System Coherence: Coherence between the CPU and GPU. (MICRO 2013)
    - Problem: The CPU directory cannot support the traffic requirement of the GPU
    - Solution: Decouple permission from memory access. Track permission on a coarse granularity, and once the GPU has permission to access a region of memory, issue memory requests directly to the memory controller.
    - This was the first paper to show a way to provide coherence between the CPU and integrated GPU, something that is now common in many shipping products.

        - I filed a patent on this, too.
- Supporting x86-64 Address Translation for 100s of GPU Lanes (HPCA 2014)
    - Problem: We want "pointer-is-a-pointer" semantics between the CPU and GPU. This requires the GPU to use the same virtual addresses as the CPU. How can you design an address translation mechanism to support the high-bandwidth GPU?
    - Solution: Use the inherent architecture of the GPU to filter most of the accesses to the TLB. However, we still need to provide high translation bandwidth via multi-threaded page table walkers.
    - This was the first paper (concurrently with other work out of Rutgers) to look at GPU address translation.

Improving the efficiency of analytic database workloads

- Problem: I recognized that analytic database workloads are bandwidth-bound. They consume large amounts of data (many GB to TB) and are increasingly performance-driven as users want results in "real-time" (e.g., 10s of milliseconds)
    - This is a growing application domain. Many of the computers in datacenters run this kind of application. So, it's important for it to be high-performance, low-power, and cost-effective.
    - Industry sees this as an important problem. Oracle's DB accelerator targets this workload, and there are many products specifically targeting this domain (SAP HANA, Oracle Exadata, IBM Blu, Vectorwise, etc.)
- Modified scan and aggregate algorithms for the integrated GPU (DaMoN workshop 2015 with SIGMOD)
    - Problem: CPUs are overprovisioned for this kind of workload. It is likely data parallel hardware (e.g., GPUs) can be more energy efficient.
    - Scan solution: Apply the bitweaving algorithm to the very wide vector architecture of the GPU.
    - Aggregate solution: Leverage the tightly-integrated GPU by using the CPU for the less parallel part of the computation
    - Takeaways:
        - Discrete GPUs waste 95% of their execution time in data transfer
        - Integrated GPUs perform slightly better than multicore CPU with same memory
        - Integrated GPUs use less power (and thus save energy) than a multicore CPU
- Implications of 3D die-stacked DRAM (SIGMOD Record journal)
    - Problem: New technology promises much higher bandwidth (e.g., 3D-die stacking). How can we take advantage of this bandwidth?
    - Using an analytic model to predict performance, I found
        - CPUs do not have enough memory-level parallelism to exploit 100s GB/s of bandwidth, but integrated GPUs do.
        - Integrated GPUs' power benefits over the multicore CPU are much bigger when using high-bandwidth memory
- Analytic database appliance (Big-data workshop (BPOE) with ASPLOS 2016)
    - Problem: How do you design a system given CPUs, GPUs, DRAM DIMMs, and 3D die-stacked memory if it is only going to perform real-time analytic workloads (bandwidth-bound computations)

    - I further developed an analytic model to project the power-capacity-performance tradeoffs between these systems.
    - Takeaway: Die-stacked memory makes sense only when performance is paramount. If performance doesn't matter, the power costs and limited capacity of die-stacked memory limit its effectiveness.

Other contributions

- I seek out collaborations
    - Worked with Jignesh Patel and Yinan Li (his student) on the database projects above
    - Worked with Lena Olson on "Border Control" which protects the system from untrusted accelerators that are logically integrated via things like my cache coherence and shared address translation work.
    - Worked with Hongil Yoon and Guri Sohi on virtual caches for GPUs (currently under submission, but very close at ASPLOS). This work tries to lower the overheads of address translation on integrated GPUs, and it plays well into my previous work.
- Teaching
    - I really enjoy teaching and took most opportunities I had to teach. I taught a few lecture here and there for Mark and David when they were sick or out of town.
    - I taught an undergrad lecture course: CS 354 Machine Organization and Programming.
        - I stepped up when the previous lecturer left less than 2 weeks before the start of the semester.
        - I taught 141 3 times weekly, but I developed the lesson plans for both lectures (total of about 250 students).
        - Received applause after my final lecture, and got very positive feedback on my evaluations like "Jason's teaching style is the most engaging and interactive that I've ever seen in a lecture section. He takes time to answer students' questions and allows them to take part in the material."
    - Volunteered at a Madison elementary school to teach an after school CS class. Here I got to engage with the next generation of diverse computer scientists.
- gem5 community leadership
    - I've been pushing the community hard for better documentation. I've been working on a book (slowly) called learning gem5 (learning.gem5.org). I'm running a tutorial at the upcoming HPCA to introduce people to gem5.
    - I've made significant contributions both to the code base and the community as a whole.
    - As others have decreased their activity in the community, I have stepped up to take on more leadership
- I say "yes". I go out of my way to do things that benefit the community
    - I have reviewed a number of people's papers prior to submission to help them out. E.g., Mike's recent ASPLOS paper that was just accepted and I've reviewed a number of Karu's paper and anyone from multifacet who's asked me to.
    - I created the computer architecture @ UW poster

- I have reviewed papers for VLSI, HPCA, DaMoN, TACO, and TODAES