Group #11: Community of Interest on the Future of Scientific Methodologies

|  |  |
| --- | --- |
| Date | November 2, 2020 |

|  |  |
| --- | --- |
| Participants | |
| Scott Pakin | Nhan Tran |
| Neil T | Javier Duarte |
| Bruce | \*fac- Nami |
| Dan Stanzione | Katie Schuman |
| Prasanna Balaprakash | Martin Foltin |
| Yasaman Ghadar | Grace Johnson |

Contents

[1 Day One - November 2, 2020 3](#_Toc55994524)

[1.1 Discussion: Breakout 1 - Define the Scope of the Problem. 3](#_Toc55994525)

[1.2 Discussion: Breakout 2 - Implications of this Problem. 6](#_Toc55994526)

[1.3 Brainstorm: Day 1 Reflections 9](#_Toc55994527)

[2 Day Two - November 5, 2020 10](#_Toc55994528)

[2.1 Brainstorm: Breakout 3 - Signposts 10](#_Toc55994529)

[2.2 Brainstorm: Breakout 4 - Signpost Plausibility 12](#_Toc55994530)

[3 Day Three - November 10, 2020 15](#_Toc55994531)

[3.1 Brainstorm: Breakout 5 - Pitfalls and Roadblocks 15](#_Toc55994532)

[3.2 Presentation: Breakout 5 Figure 18](#_Toc55994533)

[3.3 Brainstorm: Breakout 6 - Keys to Success 18](#_Toc55994534)

[Appendix 20](#_Toc55994535)

[Live chat 20](#_Toc55994536)

1 Day One - November 2, 2020

1.1 Breakout 1 - Define the Scope of the Problem.

Participants: 0

**Question or instruction for the discussion:**  
Breakout 1 - Define the Scope of the Problem.  
The purpose of this session is to lay the foundation for the next 5 sessions. That is, each breakout group will define a key piece of technology, a new device, or methodology that would have an impact on how the labs/scientists operate. The details should include answers to the questions below.



**Sticky points:**

 Top Takeaways (5 points per participant)

* What is the problem, issue, technology, device, methodology?
  + (1) Multiple co-existing computing paradigms: What is the development and delivery model for future computing facilities to deliver classical, quantum, neuromorphic computing and other specialized or highly targeted computing systems. (#1)
  + (1) "Paradigms" aren't going to be strictly about hardware shifts -- I think we are also seeing a shift from "batch" to other modes of computing use -- interactive, real-time, on-demand, driven by human or as services/API. Particularly as our hardware becomes more heterogeneous, can we focus on shared systems and utilization, or do we need to change our mode and metrics? (#2)
  + (1) Future high performance computing facilities will be extremely heterogeneous and will include neuromorphic, quantum, and likely other specialized systems. Should future facilities be centrally located as in today's leadership computing facilities or should they be more dispersed -- should we move to more of a cloud model for some of these future systems? (#3)
    - Arguably, centralized systems \*are\* a cloud model -- I think maybe the key difference is they are overprovisioned so you can grab partitions of hardware that are usually available, not busy. (#5)
      * Agreed, but I think the ability to "grab partitions" will be even more important in these extremely heterogeneous systems. We need the ability to share these more effectively than what's typically available today. (#6)
        + Yup -- I agree -- so is changing metrics/scheduling/co-scheduling what we need to do? (#18)
    - Also, how would we manage memory movement? Could we consider moving compute to memory instead of memory to compute? (#19)
  + (4) The big challenge is that computer hardware is fracturing because of technical and economic forces. As a result, we will increasingly have accelerators of various kinds (with many different architectural and substrates). The two big questions facing DOE will be: (i) how to build a portfolio of computing architectures that meets the department / government / nation's needs, and (ii) what role DOE should have in shaping the portfolio of architectures that are being developed? (#4)
    - Along with the fracturing of the hardware base, the programming model may fracture too -- and that would almost undoubtedly be bad for science - what is the model to exploit all this will be important. (#7)
      * Agreed. This could fragment all the way up to software and APIs, for example as google did with TPUs and tensorflow. (#11)
        + (4) How do we make hardware both accessible and abstracted? As experiments take more and more data, experimental design demands integration of the hardware as close as possible to data creation. How do we open that up to the domain scientist? (#8)

(1) Future computing systems will be increasingly heterogeneous, which means specialized and potential higher barriers to usability. We should consider some sort of unified interface where scientists can use available hardware efficiently (time-wise and FLOPS-wise) (#13)

(2) How can anyone program a system made up of a large number of conceptually different components? There's a huge gap between today's CPU+GPU and tomorrow's neuromorphic+quantum+chemical transport+membrane+DNA computing+... (#10)

+1 on this -- successful abstractions haven't exactly been our strong suit! (#9)

+1 -- And we need to start focusing on this now! We can't wait for the hardware to be fully developed to work on abstractions and accessibility. (#14)

* + (5) What role will/should AI/ML play in the programming model of these systems to help with usability of these systems? (#12)
    - Code generation seems likely t0 be a bigger part of far-future applications. (#15)
  + Many future technologies are non-deterministic? How will we know if we received a correct result from such computer systems? (#16)
  + (2) What is the way we should conceptualize different computing paradigms? Are there common denominators which can be as "future-proof" as possible (acknowledging this is impossible)? (#17)
  + What is the problem? (#21)
    - Problem: Availability of very specialized, very esoteric hardware that is hard to exploit both individually and in concert with each other. (#24)
    - (1) Need to understand what are the computational building blocks (i.e. computational motifs) that will be useful to users and also amenable to hardware / algorithm re-designs that accelerate the computation. These will then need to be orchestrated into cohesive workflows for particular domains, e.g. weather simulation, either by humans or automatically by AI. (#26)
      * and how do we separate that from the hardware vendors and proprietary software stacks? (#27)
        + The "vendor-specific ecosystems" are about to become a huge problem. Let alone CPU-GPU, we won't be able to make software work across the 3 different brands of GPUs. At least not work well. (#35)
    - How do we train non-experts to use these computational blocks? (#30)
* Who would develop it (basic research to advanced deployment)?
  + (1) For developing hardware portfolios, the management of the supercomputers should aggregate up the usage of various computational motifs so we know which architectures need to be built. (#20)
    - I'm a little worried that if we do it in this way, we will become HPC centric and lose the edge/IoT application space (but I may be misunderstanding the comment) - Nhan (#44)
    - We also need to develop an ecosystem and programming paradigm for how all these computational motifs talk to each other. Can we string different hardware together? How do we express that? (#46)
  + (2) There needs to be significant communication in addressing this problem across the compute stack, from hardware developers to software and algorithms developers to application developers. We cannot do this R&D in isolation from one another. (#23)
  + Industry tends to focus on its one "value added" piece. Perhaps DOE can be tasked with integrating concepts across vendors into a form that is usable by domain scientists, even if this unification will not be a money-making enterprise. (#31)
    - +1 -- I think DOE can help play a role that crosses vendors \_and\_ disciplines to find the commonalities across application spaces. (#34)
      * Another +1 here. (#37)
    - +1! (#40)
  + Basic research is needed both for the individual components working in isolation and the grand vision for how to combine disparate elements into a unified computing environment. (#29)
* Who would use it and what skills would they need to use it effectively?
  + (1) Domain scientists from various disciplines. Ideally, they will not have to learn specialized languages for each component-this will be managed by a higher level language (potentially including an AI) (#22)
    - Do we expect a common higher level language? Or will domain-specific languages be more common? (#28)
      * I would think some sort of common language (or AI system) would be needed. I can't imagine many users being willing to learn dozens of new DSLs, one for each tiny piece of their problem. (#33)
      * My intuition would be the opposite. Higher level languages are usually less efficient, so I think we'll see more and more domain-specific languages. To manage the programming complexity of these, they might be done at the application domain level rather than the hardware level, so individual programmers don't have to learn too many. (#36)
        + +1 -- I expect application-domain specific languages are much more likely to emerge. (#39)

+1 I also expect to see more DSLs -- getting small communities to adopt and support them is another matter though (is there really a sufficient market for a tectonic plate-specific language?) (#41)

* + - * + Good point. I was thinking of lower-level DSLs, but you're right; if the domain really is sufficiently high-level, and the implementation can hide the hardware from the programmer, then this is the way to go. (#42)
        + In my domain, I have seen enthusiasm for wanting to \_understand\_ the benefits of different hardware, but still have an abstraction high level language layer. We need to train people in how to translate between the two (#47)
  + We might want to discuss the relation of scientific and non-scientific users of these year-2040 (or whatever) computers. DOE tends not to be well aligned with what the rest of the HPC community is doing. (How many AWS users even know what a batch queue is or how to parallelize with MPI?) (#38)
    - On the AWS question, the answer is almost 0% -- but that's not the broader HPC community. MPI is actually pretty common across the academic (and even commercial) HPC community, but the DOE ECP is moving further from standard practice. (#43)
* When would it be expected to be in production use (N years in the future)?
  + The specialized chips in the portfolio are already being added (GPUs), but will increasingly appear over the coming decades. Substantial work will be needed to develop the software / OS architecture to make managing these easy and efficient. (#32)
    - Hardware will (likely) be increasingly heterogeneous - even at the chip level (see the current explosion in new processors) -- the programming model and "killer app" will determine adoption -- GPUs for general purpose have been widely discussed since 2003, and arguably hit their stride ~15 years later -- after an awful lot of effort and spending. Can this happen for 10 architectures? Most will \*never\* make production. (#45)
* Where, and how widely, would it be deployed?
* What is the setup time and/or process for using it?

1.2 Breakout 2 - Implications of this Problem.

Participants: 0

**Question or instruction for the discussion:**  
Breakout 2 - Implications of this Problem.  
Each group will now develop a list of issues and implications for the issue/technology/community they settled on. There are lots of implications for how a technology can be used, or further developed.



* What other/companion technologies, services, software/hardware must also be developed and deployed?
  + Communication technologies. If we're going to have lots of processors communicating, we don't want the network to devour all of the performance we would hope to see. (#6)
    - Can we reconfigure so we move compute to memory instead of moving the memory to compute? (#25)
  + AI driven program generation and co-design (#7)
    - +1 (#21)
  + Hardware design tools that take a programming motif and create a customized, high-performing ASIC. (#8)
    - Hardware/software co-design methodologies and tools (#5)
    - And also better FPGA tools when ASICs aren't realistic for the problem at hand. (#11)
  + ideally: a hardware-agnostic workload scheduling interface which goes across the whole stack from the front-end data acquisition to the data center (#9)
  + In breakout one we come up with a very heterogeneous architecture that user will have access to. In order to accomplish that, vendors creating hardware and software should come in agreement. There should be some senses of overarching goals. (#12)
  + automated code validation and verification (#13)
    - This means the user/coder gives test cases -- concerned we will never get users/coders to actually do this. It's like trying to get them to tag files with metadata -- 25 years of futility. (#23)
  + Analysis tools, akin to performance monitoring ones, that help determine which motifs are (or could be) in programs and thus help the programmer target those. (#14)
    - +1 -- these are critically important. (#26)
  + Brain plug-ins so we can wrap our heads around the huge increases in complexity. ;-) (#15)
    - +1 (#16)
    - I'll take it! Let's make it the neuroscientists problem. (#24)
  + Dynamic workload analysis and load balancing across heterogeneous architectures, perhaps via AI/ML (#17)
  + Higher-level language layer which enables a domain scientist to write efficient code for heterogeneous architectures but not have to know how to program on each different device. (#22)
    - tools that enable application scientists to write code on new hardware faster with maximum performance (#10)
    - Tools that allow a high-level description of a program (e.g. matrix sizes, etc.) to be analyzed to determine which are the most important computational motifs, and thus the hardware that it should be run on. (#4)
    - Open-source, high-level, user-friendly programming tools targetting different heterogeneous hardware (#3)
    - tools for making programming easy (#2)
    - Platform-agnostic, declarative programming languages with different backends to support different hardware (like OpenCL but better) (#18)
    - intelligent {programming models, compilers, OS/R, communication subsystems, I/O} (#1)
    - automated program translation from one model to another model (#19)
      * Any evidence that this can actually happen? Models are almost by nature not equivalent. (#20)
* Who is/will develop this companion technology/service?
  + DOE scientific computing leadership needs to lead the strategic process to (i) understand which motifs are most important for DOE workloads, and (ii) decide which hardware architectures will need to be part of the DOE hardware portfolio to meet these needs (including HPC, sensors, etc.) (#27)
    - DOE should be coordinating -- vendors will likely be developing some (or all) of the hardware, but the applications will be on the DOE side. (#28)
      * DOE's budget and chip demands are already sufficient to be economically having chips designed just for them. (#35)
    - I pushed back on this a little bit in another thread so I'll add my name here for culpability :) -- I think DOE (DARPA/NSF/...) should take some leadership here, but it goes beyond the supercomputers and solutions for supercomputers might be restrictive. [Nhan] (#39)
  + industry/vendor but with the help of DOE (we need to influence them; otherwise commercial market will not dictate what we really want) (#29)
  + A lot of the innovations in programming models and applications and techniques come from the university space -- but DOE can do an awful lot to make reasonable choices from this broad set of prototypes, and make them widely accepted. Vendors can be trusted to do what is best for vendors, and won't be interoperable unless we make them. (#30)
  + Need DOE/industry partnership (infrastructure already in place) (#31)
  + I would like that universities and research institute to be in charge of what they need and they communicate that to DOE for the next computing systems. (#32)
  + At first, everybody. DOE, industry, universities. Lots of tries, then DOE coordinate with everyone to take the best features of the initial attempts and unify them (#33)
    - This could be done through smaller DOE funding calls; then a pruning/merging to unify efforts (#42)
  + There is clearly a tremendous amount of computer science research as well, so this also requires collaborations from universities, DOE research organizations, DOE facilities, and vendors. (#34)
  + Performance tools can come from anywhere. The developers of the hardware can be expected to provide tools for their piece. However, once we get into high-level DSLs, the developers of those DSLs (most likely DOE and/or academia) would probably create the associated performance tools. (#36)
  + By contrast -- only the vendors can make the silicon, but it helps a lot when big entities like DOE can speak with a unified voice. (#37)
  + We need a mechanism to productize prototypes at the Uni/DOE space and see it in vendor software stack (#38)
  + Academia is arguably stronger in compiler technology than DOE but tends to have less HPC acumen. DOE and academia would need to work together on that front. (#40)
  + Establishing standards for hardware heterogeneity committee to make sure we have a say (hardware space is like what MPI looked like several decades ago; MPI standards committee is a good model to unify) (#41)
  + DOE labs as testbed (#43)
* What skills/knowledge does the end user require?
  + I think the goal is to try to reduce the burden on what we require the end user to know -- via layers of abstraction in the programming model, but also probably AI/ML tools to help the user in developing their code and mapping their code to extremely heterogeneous compute. (#44)
  + It's crucially important that we if we \*add\* new things that end users need to know, we \*remove\* some other things that by hiding them in the abstractions. People will have very different skills in 2040, but the capacity to learn things won't be a lot higher. (#45)
  + Users will need an open mind and a willingness to attempt to do things differently from how things have traditionally been done in their field. (#46)
    - I both agree, and think this runs counter to human nature. :). (#52)
  + Performance engineering spellcheck: AI code mentors that look at the code that you've written and suggest changes that would make it run more efficiently on the available hardware platforms. (#47)
    - +1 Love this idea. (#49)
      * Autocomplete but for your code (#51)
    - like Clippy from Microsoft Word (#53)
      * :( (#60)
      * RIP (#62)
  + If the domain user could learn how to express/translate their task into "common language blocks", that would go far in meeting software/hardware in the middle (#48)
    - And build these into domain specific languages, maybe with an eye to being able to translate them across hardware architectures. (#54)
      * +1 (#57)
    - Need to develop tools that minimize the requirement of skills/knowledge (#50)
    - Tools for programming multiple systems with minimal effort (#56)
      * But also testing multiple configurations under the hood to ensure the code efficiently uses the hardware (#59)
    - +1!! (#55)
  + While I agree it's good to abstract unimportant things away, I think there also needs to be some education into different hardware architectures and how the code needs to be optimized for those (#58)
    - We probably want to have multiple layers of abstraction and allow the user to decide how close to the hardware they feel comfortable. (#61)
* What are the training/support requirements?
  + Training on tools..when to use what ...where to enter the hierarchy (as Katie mentioned) (#63)
  + Vast -- Quantum means we needs to largely rethink our notion of "programming". (#64)
    - Reshaping our "programming" curricula -- what programming means changes, so computing education (from secondary school on) needs to be updated to reflect that. (#65)
    - Even with current ML systems, "programming" sometimes comes in the form of "training". (#66)
  + Understanding/training for tradeoff between prototyping vs productivity vs performance (#67)
    - +1 -- Helping users understand when it is useful to spend time optimizing their code and customizing to the underlying hardware. (#70)
  + Teach users about large scale programming motifs that correspond to optimized computational silos (#68)
  + For the DOE (or whatever organization): it's really important to develop a library of use-cases that go beyond typical stove-pipe boundaries to develop the tools we are dreaming about. This includes open datasets/challenges and multidisciplinary teams with different levels of expertise (#69)
    - (and just to add, I think that support for these mechanisms are growing in availability) (#74)
  + Targeted workshops on programming heterogeneous hardware (#71)
  + Hardware-software design courses as part of science graduate education (e.g. one on FPGAs in CSE at UCSD: https://pp4fpgas.readthedocs.io/en/latest/) (#73)
    - Provide resources to universities to train the next generation of computational scientist with related computer science informations. (#72)
      * Increased awareness of DSL-type ideas in general scientific/computational science training (#75)
    - lab/university partnerships for training (#77)
    - +1 (#76)

1.3 Day 1 Reflections

Participants: 0

**Brainstorm question or instruction:**  
Day 1 Reflections  
This area is for the Moderator to note key discussion points to summarize what was accomplished in Day one. Remember that day one is focused on Identifying a new technology or methodology and identifying the implications and possible consequences of it. The moderator can populate this individually at the end of the day or request input from the group here.



* 1. Computing is hitting limits as we want to scale up computing power but are faced with rising costs and energy usage. To fix this will require us to move to specialized chips, which will be more efficient, but will also introduce complexity in hardware and software.
* 2. What I expect in 30 years: A computing ecosystem with a vast array of hardware types and an associated software stack that enables ease of programmability and usability of the diverse hardware backends.
* 3. Movement to other computing platforms may enable new algorithms and approaches that provide vast speed-ups, which could be transformative in the types of computations that can feasibly be done.
* 4. New hardware and computing paradigms require new programming tools and knowledge to be developed and propagated throughout the scientific community
* 5. in this group we talked about how we should think about computing resources that look at different types of problems and different sizes. One idea was to create a complex machine made of CPU, GPU, Quantum computing, AI and NN and so on. We talked about how instead of writing new codes have AI distinguish the type of resource based on the needs of the code. We talked about computational scientist needs to get exposed to compute science more and more.
* 6. Software is a huge issue. We have enough trouble programming today's heterogeneous CPU+GPU systems. How will we program radically different hardware components?
* 7. Specialized computing (neuromorphic, quantum, DNA) may change our concept of what computing is. This is worth something in it's own right (aside from applications, computing efficiency)
* 8. New hardware and computing stacks will be required to enable continued performance improvements in computing, which will enable us to scale modeling and simulation to even greater scales AND address challenges associated with scientific data, which will continue to grow exponentially both in scale and in complexity.
* 9. Why? Advancing science is predicated on evaluating hypotheses based on (increasingly complex and overwhelming) data. Accelerating science means accelerating hypothesis generating and testing by processing/interpreting our data as efficiently as possible. Efficiency requires finding both the right tool for the job and the right formulation of the question. Our goal is to impedance \*match domain formulations of questions with the right computing tool\* to accelerate science discoveries by orders of magnitude

2 Day Two - November 5, 2020

2.1 Breakout 3 - Signposts

Participants: 0

**Brainstorm question or instruction:**  
Breakout 3 - Signposts  
What we are looking for is technology or social trends that would give us clues that we are on the right track. o How would precursor technologies/services be identified? o What are the precursor technologies/services? o Is there a rank order for when specific technologies/services need to be available? o What DOE or Lab policies need to be in place now, in 5 years? o What facilities need to be in place now, in 5 years?



**Sticky points:**

 Top Takeaways (5 points per participant)

* 1. topic 1: How would precursor technologies/services be identified? In this group we came to agreement that we are taking in to consideration any computing resources called "computing echo system" that could include: HPC, leadership computing facility, laptop, desktop, mobile phones and so on.
  + Comments
  + New approaches to programming (programming by design/optimization) (#2)
  + Hardware-aware compiler (#3)
    - AI/ML hardware-aware compiler as a precursor to this precursor (#14)
      * Build a challenge/data set which requires multiple computing paradigms (#17)
        + Define a computing ecosystem which can plug in multiple heterogeneous compute paradigms for AI/ML methods (#18)

We also need the actual heterogeneous platform testbed to test it on -- ideally in an integrated environment and not in disparate systems. (#23)

This sort of testbed/ecosystem needs to be adaptable and evolvable as new hardware emerges. (#27)

For any heterogeneous system, we need to define interface/communication protocols (#25)

* + Self adaptive and self tuning OS/R (#4)
  + Prototype extremely heterogeneous systems to begin to evaluate on, beyond just CPU/GPU-based systems. (#5)
    - Testbeds that can help us identify what hurdles will occur in the infrastructure and usability of these extremely heterogeneous systems (which are likely to have very different usage characteristics than today's HPC systems). (#21)
  + Automate application development (DSL based, composable and components) (#6)
  + Assess a representative sample of existing DOE computing loads to determine the computational motifs that are being used. This can then be used to determine hardware portfolio needs. (#7)
    - AI/ML approaches for identifying commonalities in computing loads (#9)
    - Prototype a computational motif analysis pipeline, where a existing DOE workload is analyzed to distill which motifs are being used (or could be used) (#26)
    - Map computational motifs to existing hardware options (#28)
  + something like Android on steroids for HPC: OS/R that runs on any hardware but hardware-specific customization should be automatic (#8)
  + Identifing computational motif to best utilize the resources. (#10)
    - perhaps assisted by AI (#22)
  + Miniaturization of components and reduced power consumption. DOE should keep its eyes on technologies that are currently physically large (say, machine-room-sized) that are shrinking in footprint. Think of how mainframes shrank to fit in a cell phone and smart watch and how that changed what society was able to do with them. Current quantum computers require lots of care and feeding, but imagine if you could carry one in your pocket. (#11)
  + Technologies getting cheaper and omnipresent (#12)
  + Ability to precisely control natural phenomena. Feynman proposed quantum computing in the context of using programmable quantum systems, rather than classical simulation, to study quantum phenomena. One can imagine remotely 3-D printing an object and placing it in a physical wind tunnel to study airflow. ColdQuanta lets users create and measure Bose-Einstein condensates from their desktop. What other experiments can be made "programmable"? (#13)
    - Terminology: "analogous computing" (#24)
  + Find new computational models that are physical systems that we can control, but which are analogous to a broader set of problems that we care about. For example, quantum systems help us solve a broad range of atomic interaction issues. But there may be many other physical analogs that we could take advantage of. (#15)
    - Think: Berkeley motifs, but perhaps at a larger granularity (#16)
    - Great point. We could end up with motif patterns at many different levels (small kernels -> whole computational patterns) (#20)
  + flexible application (for ex JIT based on the (temporal) hardware availability) and configurable hardware that adapt based on the application code (#19)
* 29. topic 3: ranking technologies/services in time
  + Comments
  + 5 year horizon (#31)
    - (4) DOE-wide test bed (#33)
      * That is flexible and evolvable as new hardware is developed (#46)
        + (2) A rapid-prototyping capability. The ability to create hardware proofs of concept before embarking on big follow-ons. (#51)
        + (1) standardize and test interface/communication protocols between devices (#52)
    - enable rapid collaboration between labs/experts/univ (#35)
    - (2) multiple hardware specific AI/ML compilers (#36)
      * (implying that then we have to stitch them together) (#38)
        + (1) build hardware and hardware simulators (#34)
    - (1) Common benchmarks and evaluation metrics (#44)
      * (1) (multiple) grand challenges for systems that requires heterogeneous heterogeity (#48)
    - (4) Identify computational motifs (#47)
      * Tools to automatically identify computational motifs (#49)
  + Longer-term (#32)
    - (2) AI/ML code spell-checker that suggests improvements to make code run faster (#37)
    - (2) Dynamic application analysis (perhaps via AI/ML) to map to hardware most effectively for a given application needs. (#41)
      * Needs to be able to evaluate code running-time on different hardware (#42)
    - (2) Portfolio of specialized hardware architectures and substrates that fit DOE workloads (#50)
    - (2) Tools to democratize the technology and make sure hardware vendors agree to common goals (#54)
    - Allow definition of computational motifs to evolve flexibly (and more abstractly) such that it can incorporate emerging technologies (#56)
    - (2) Develop an analogous-computing capability for science of interest to DOE (#59)
* 30. What DOE or Lab policies need to be in place now, in 5 years? What facilities need to be in place now, in 5 years?
  + Comments
  + DOE needs to be more willing to fund radical new ideas that haven't yet caught on in industry and that are higher-risk, higher-reward than what they're used to. Currently, DOE says, "Hey, ML/AI are big. Let's fund lots of ML/AI." and "Ooh, everyone's talking about quantum. Let's fund quantum." But if a researcher currently wants support for some novel technology that no one's heard of, good luck. (#40)
    - DOE should also not be afraid of hardware prototype development -- as we move towards more and more specialization, we may not be able to rely on vendors exclusively anymore. (#53)
  + Small funding resources for building testbeds. (#43)
    - Policies that facilitate ease of access to these testbeds across labs. (#55)
  + Have programs that encourage collaboration with industry in smaller scale for rapid results. (#45)
  + On-going analysis of the computational motifs being used and how well they fit DOE's existing hardware portfolios (#57)
  + Co-design labs with low overhead to prototype (#58)
  + enable rapid collaboration between labs/experts/univ (#60)

2.2 Breakout 4 - Signpost Plausibility

Participants: 0

**Brainstorm question or instruction:**  
Breakout 4 - Signpost Plausibility  
Now that we have the list of signposts, the groups need to consider how plausible they are and what DOE needs to do to either ensure they happen or the implications of them not happening. o Who is actively working on these precursors? o When would these precursor technologies/services be needed? o What active or pending research programs need to be in place now? In 5 years? 10? o What existing or planned facilities need to be in place now? In 5 years? 10? o What software services or capabilities need to be in place now? In 5 years? 10? o How successful has the community been in meeting previous goals?



**Sticky points:**

 Top Takeaways (5 points per participant)

* 1. a) DOE-wide testbeds
  + Comments
  + Who is actively working on this? (#8)
    - Quantum testbeds are being built at a few labs (LBNL, ORNL, and partners) as part of an ASCR quantum-testbed grant. LANL makes available access to its D-Wave quantum annealer. (#10)
    - There are isolated experimental hardware testbeds probably at most (if not all) national labs and at many universities, but they're relatively loosely coupled hardware systems and not widely accessible. (#9)
    - No unified (national-level) infrastructure; (#36)
  + When would be needed? (#11)
    - These are needed as soon as possible. (#16)
      * +1 (#17)
  + What research programs need to be in place? Now, 5, 10 years? The panelist indicated that these research programs needs to be in place now and continue to grow. (#12)
    - Exploratory research to identify promising computing paradigms that are off the beaten track. These should start as soon as possible because they may have a long lead time to determine their practicality. (#18)
    - Now (#22)
    - We need research programs NOW to investigate the development of the software ecosystem for these co-existing computing paradigms. (#24)
    - A research program which incentivizes development of hardware for science (not industry hardware build for industry problems) or a way to work with industry to develop with them from the start (#26)
  + What existing or planned facilities need to be in place? Now, 5, 10 (#13)
    - Hardware design is not traditionally one of DOE's strengths. It would take a long time to build up the facilities needed and staff them appropriately. (#25)
      * This could also be carried out via partnerships with industry. (#27)
    - Start with specialized testbeds focused on aspects (e.g. software, networking, similar types of hardware,...) (#28)
  + What software services and capabilities need to be in place? (#14)
    - A software ecosystem that makes usability easier for plugging in different hardware devices (#19)
* 2. b) benchmarks and test suits to evaluate the systems
  + Comments
  + Who is working on these? (#29)
    - MLPerf is a nice organization which sets up benchmarks and challenges. However, it's less "R&D" focused -- it would be nice to set up an extension or clone of this for DOE science challenges (#38)
    - ECP provides applications and proxy apps that represent computational science of interest to DOE (#40)
    - ECP also has a continuous-integration effort that enables applications to be tested easily across DOE sites as part of routine code development. (#41)
  + When would we need them by? (#30)
    - Now, since there are ongoing efforts in these areas and requires less financial investments. (#44)
  + What research programs need to be in place? (#31)
    - We need a research program just to define what the benchmark and test suites should be. This sort of program should include the application domains (to motivate the benchmarks and test suites) as well as the software/hardware developers. (#43)
    - We should work defining a rigorous (or at least repeatable) program for measuring system usability (#69)
      * Surveys to users, quantify man-hours for on-boarding. Start quantifying so we can evaluate as we shift to more automatic on-boarding -- as systems get more heterogeneous it won't be possible to assist every user with on-boarding manually (#82)
  + What existing or planned facilities need to be in place? (#33)
    - maybe take advantage of DOE research programs rather than facility at this time (#46)
  + What software services need to be in place? (#34)
    - infrastructures needed for CI, measurements, validation, etc... (#42)
    - performance tools for measurement (#45)
* 3. c) determining computational motifs
  + Comments
  + Who is working on this? (#15)
    - UC Berkeley: Phil Colella, Old ParLab group (#5)
    - Compiler designers / Performance engineers (#7)
    - MIT: Albert Reuther (Lincoln) (#6)
    - Neil Thompson (MIT): Project on the Computation Motifs being used by 40 of the U.S.' most advanced computing users (#52)
      * researcher website: www.neil-t.com (#54)
  + When would technologies be needed? (#20)
    - Needed ASAP: Examine current DOE workloads and motif usage (#21)
    - Within 2-3 years: Strategic plan for hardware mix, based on DOE motif workloads (#23)
    - Within 5 years: Create first custom hardware for DOE (and others) based on DOE motifs that are underserved. (#32)
    - Within 5 years: First version of AI/ML proofreading software that tries to identify motif usage in code (#39)
    - At 5 years (and every 5 years thereafter): Strategic plan for the hardware portfolio that DOE will be buying and targeting for applications (#35)
    - 5-10 years: Comprehensive view of which motifs are available (ranging from low-level to high-level) and how programs can be adapted to them. (#37)
      * Think of motifs as layered, from low-level (e.g. dense matrix multiplication) to high-level (e.g. particular operations on images) (#64)
      * Can we build out a type of "layered" motifs with different level of abstractions from the computational architecture to the domain application abstraction (#66)
  + What active or research program need to be in place now? (#47)
    - Automatic or semi-automatic extraction of motifs (#50)
      * 3-5 years: We will need AI/ML software/algorithms to automatically identify computational motifs moving forward. (#51)
    - Characterization of motifs: Do they represent algorithmic patterns? memory patterns? instruction-mix patterns? I/O patterns? communication patterns? (#53)
    - Generalization of motifs: Just because a problem is handled one way on today's computers, how differently could this problem be solved on a radically different computational platform? (#55)
      * +1 (#67)
  + What existing or planned facilities need to be in place? (#48)
    - This effort does not require a facility (at this time) and rather a program (#62)
  + What software services or capabilities need to be in place? (#49)
    - Ready access to DOE applications and the associated support (documentation, a variety of input decks, etc.) (#56)
      * Ready access to data (code itself, documentation) from scientific (DOE) application areas -- a central repository an AI could learn on --- like www.osti.gov (#68)
      * Ready access to the domain scientists as well? (#65)
* 4. d)hardware specific co-design tools (AI/ML, traditional simulations, etc)
  + Comments
  + Who is working on this? (#57)
    - Big industry players (e.g. Google, Amazon) (#71)
    - a lot of people? especially as you get deeper into the hardware (#72)
    - ASCR is looking into this through funded programs (AI codesign) (#73)
  + When would these technologies be needed? (#58)
    - Traditional software simulators for different technologies (and the collective heterogeneous systems composed of disparate technologies) need to be developed now. (#63)
  + What active or pending research programs need to be in place now? In 5 years? 10? (#59)
    - Rapid-development tools. As a new technology emerges, we need to be able to design hardware for it, simulate it, develop software for it, ... (#70)
    - Now: Research in reprogrammable hardware systems and software stacks to enable usability of those systems (#75)
    - Multi-disciplinary grants / teams for going after the benchmarks/challenges (#81)
      * in order to build a successful co-design workflow (#83)
  + What existing or planned facilities need to be in place now? In 5 years? 10? (#60)
    - Facility for rapid prototyping of ASICs (#78)
  + What software services or capabilities need to be in place now? In 5 years? 10? (#61)
    - Program-analysis tools for quantifying the potential benefit of porting applications to new types of hardware (#80)
* 74. Cross-cutting sign posts
  + Comments
  + policies for accessibility DOE-wide (#76)
  + policies for industry collaboration (#77)
* 79. How successful has the community been in meeting previous goals?
  + Comments
  + I would say for each goal is different. For example DOE-wide testbeds, DOE has not achieved that in the scale that we discussed. As for benchmarks and hardware co-design there are ongoing effort (#84)

3 Day Three - November 10, 2020

3.1 Breakout 5 - Pitfalls and Roadblocks

**The following participants have not been active:**  
Bruce, Prasanna Balaprakash, Martin Foltin, \*fac- Nami, Neil T

**Brainstorm question or instruction:**  
Breakout 5 - Pitfalls and Roadblocks  
Detailed discussions on identifying pitfalls and potential roadblocks. If possible, list in rank ordering. o What could prevent the technology/service/device from being developed (funding, materials, policies, researchers, operations staff, etc.)? o How will progress be measured/evaluated? o How will lack of progress be measured/evaluated? o Who will decide if progress is being made? o What are the consequences of not engaging in this area?



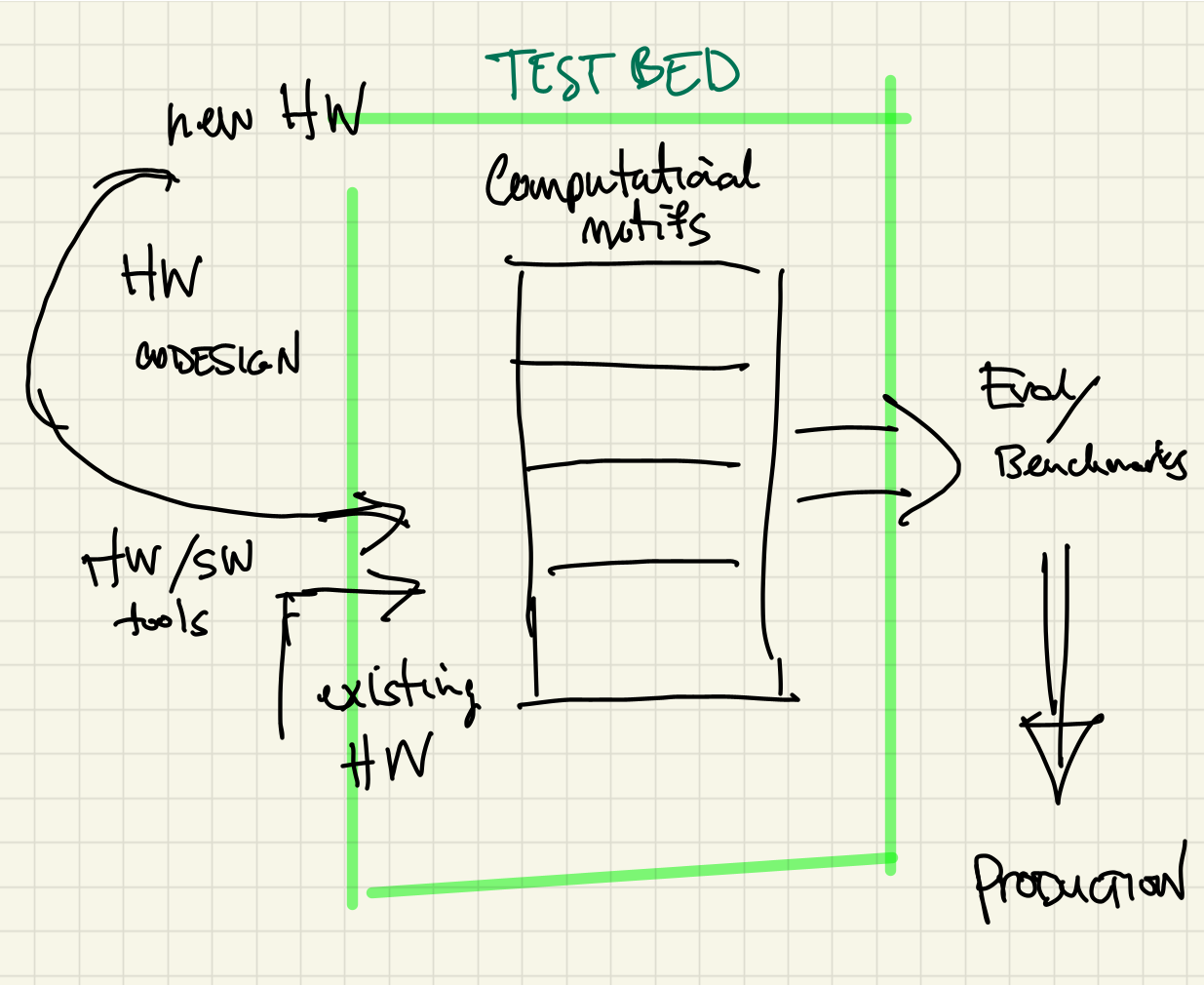
**Sticky points:**

 Top Takeaways (5 points per participant)

* 1. DOE-wide test beds
  + Merged items
  + The biggest danger in pushing forward new technologies is the "chicken and egg" problem -- No one wants to make investments until a technology is "proven", but no technology is proven until there is significant investments in programming models and adoption. (#8)
  + Comments
  + Pitfall: Individual testbeds that are not easily accessible or usable by the broader DOE community. (#6)
  + What could prevent it: Lack of cooperation with relevant vendors (#7)
  + Pitfall: lack of funding and/or organization from DOE (#9)
  + Roadblock: Industry-developed hardware may only be accessible by one lab or institution (#10)
  + Pitfall/roadblock: Proprietary software that prevents integration (#11)
  + Another danger is at the translation step -- a testbed is built, and used, but there is no clear way to translate to the "production" path. (#12)
  + Roadblock: Lack of common design tools and coordination across the labs make it hard to have a coherent development environment (#13)
  + Roadblock: software infrastructure and support is needed to make a DOE-wide testbed really accessible and usable by a broad user base (#14)
  + Pitfall: Lack of coordination amongst research efforts for disparate hardware systems (quantum, neuromorphic, etc.) that will make it difficult to define common design tools. (#15)
  + Measure/Evaluate: count the number multi-institutional collaborations (#17)
  + Pitfall: Different labs tend to have different authentication mechanisms that make integration difficult. (#18)
  + Consequences: insufficient data, will not arrive at the best solution (may only work narrowly) (#19)
  + Consequences: in the end, development ends up being much more expensive (both in personnel and resources) because it's repeated several times (#20)
  + Measure/evaluate: Number of different types of systems that are integrated in the same testbed. Number of different types of systems that are supported by common design tools. (#21)
  + Measure/evaluate: number of resulting publications and/or applications on path to production (#22)
  + Consequences: Everyone is developing in their own testbeds, developing their own software for their own applications and there is no interoperability or portability. (#23)
* 3. determining computational motifs (science specific motifs and HPC motifs (Network, MPI, I/O. Storage, Latency)
  + Comments
  + Roadblock: A common definition of "computational motifs" that's understood across domains (#24)
  + Roadblock: lack of participation from community (#25)
  + Roadblock: How do we define computational motifs? How are they specified? (#26)
  + Pitfall: lack of communication across application domains (#27)
  + Pitfall: How do we assign weights (importance) to different motifs? (#28)
  + Measure/evaluate: Number of computational motifs that capture multiple applications across domains (#29)
  + Pitfall: How can we handle applications that are so different from everything else that they don't match \_any\_ predefined set of motifs? (#30)
  + Pitfall: Do we need to consider motifs that are not just computational motifs but, say, storage motifs, communication motifs, memory motifs, etc.? (#31)
  + Measure/evaluate: determine "coverage" of motifs as a fraction of all relevant scientific domains/applications (#32)
  + Pitfall: if we are using AI to assist, it needs to be sophisticated enough to discover connections beyond what we could manually (#33)
    - Requires lots of data/collaboration across domains (#34)
  + Note: Even the numerics may differ across platforms. Consider direct vs. iterative numerical methods. The former might make more sense on one type of platform while the latter might make more sense on another. (#35)
* 2. Benchmark and test suits to evalute the systems
* 4. hardware specific co-design tools (AI/Ml, traditional simulations, etc)
  + Comments
  + Tools have the same problem as hardware -- no one wants to invest time into using them until it's clear everyone else is also willing to use them. (#16)
    - Pitfall: not investing in software tools for a hardware system because no one is \_currently\_ using it (which could be because there aren't good software tools to use it...) (#44)
  + Pitfall: Designing tools that are too specific to today's hardware (and not flexible to evolve for future, unknown hardware). (#36)
    - +1!! (#40)
  + Roadblock: Lack of data/code to train AI approaches (#37)
  + Roadblock: resources needed to explore hardware implementations (#38)
  + Evaluate/measure: tool adoption (downloads / installs / github stars) (#39)
  + Pitfall: Each hardware vendor will develop its own tools, and they not only won't interoperate, but they'll be completely different from a user perspective. (#41)
    - +1, was just typing the same! (#43)
  + Consequences of not engaging: complete lack of usability (#42)
    - or much slower adoption because tools will take decades to develop instead of months/years (#47)
  + Consequence: one common consequence endemic to this group...make too many similar things, waste time, resources, longer time to science,... (#45)
    - To play devil's advocate, this could be useful if there was some central effort to evaluate these competing approaches and choose the best/unify the best features (#46)
* 5. Cross-cutting sign posts
  + Comments
  + policies for industry and access to DOE-wide test beds (#48)
  + Pitfall: lack of collaboration between computational scientists and computer scientists (#49)
    - ...and among computational scientists and domain scientists, and among the same types of scientists from different domains. (#51)
  + roadblock: communication/stove piping barriers to prevent find common motifs (#50)
  + Pitfall: Existing funding models/proposal calls will restrict and inhibit the large-scale collaboration required for this (#52)

3.2 Breakout 5 Figure

* Slide 1



3.3 Breakout 6 - Keys to Success

**The following participants have not been active:**  
Bruce, Martin Foltin, \*fac- Nami, Dan Stanzione, Neil T

**Brainstorm question or instruction:**  
Breakout 6 - Keys to Success  
Identify who needs to be engaged, research communities, domain science communities, staff, management. Identify needed skills and knowledge (give examples) o What benefits would society obtain? o What benefits would the science/research community obtain? o What research communities need to be involved? o What domain science communities need to be involved? o What staff and management communities need to be involved? o What kind of management structure is required? o How broadly will this impact society and/or the science community?



**Sticky points:**

 Top Takeaways (5 points per participant)

* 1. Skills needed
  + Comments
  + DOE-wide test beds (#6)
    - Hardware/software engineering team (#8)
  + We need hardware developers, computer scientists, computational scientists and domain scientists to be involved. This will span industry, academia, and government. (#7)
    - IT, software developers too (#15)
  + application developers with incentives to use novel/emerging hardware (#9)
  + Benchmarking experts (#10)
  + Automated testing/benchmarking infrastructure (CI) experts (#12)
  + Program managers that can coordinate activities across disciplines (computing, engineering, computational scientists, domain scientists) (#13)
  + Cyber security experts (#16)
  + Networking, storage, and memory experts, to figure out how those will change in an extremely heterogeneous environment and implement them (#17)
  + AI researchers to help develop the tools required for automated code completion and all other "intelligent" software (#18)
  + Skills needed: Cross-domain communication skills -- to bridge the gaps between hardware/software/applications (#19)
    - there is a hopefully synergistic relationship between cross-domain communication skills and computational/application motifs (#22)
  + A knack for finding killer apps for exotic new technologies, or, in general, being able to identify uses for unusual computational paradigms. (#20)
  + Focussed community engagements through workshops, tutorials (bring stakeholders frequently) (#21)
  + Explicitly avoiding workshops that are targeted towards one particular hardware platform. (#23)
  + Programming languages experts (#24)
  + having training committee (teaching committee) to enable all these. (#25)
  + Needed today: Skills in scientific software best practices -- that will produce software that we can use to learn from for computational motifs, etc. (#26)
  + extracting benchmarks from applications (manual/automatic) (#27)
  + Skills needed: Ability to write modular, reconfigurable code that can be relatively easily ported to future architectures/programming models even if those are not known at the time (#28)
* 2. Benefits to society
  + Comments
  + More accessibility and usability of future computing systems, allowing for performance improvements on applications, accelerating scientific innovation. (#5)
    - +1. That's what I would have written. (#14)
    - +1, with less reliance on industry-specific tools (#30)
  + innovative use of future computing beyond DOE (#31)
  + Open-source tools will increase access to new computing systems (#32)
  + democratization of computing: what if cutting-edge computing is free (as in free-software) --> democratization of science (#33)
    - Open hardware too (#35)
      * +1! (#44)
* 3. Benefits to science community
  + Comments
  + Productivity and performance benefits for computational science, implying reduced time to scientific results. (#11)
  + Increased collaboration across disciplines -> more innovation (#34)
    - Discovering synergies via computational motifs -> breakthroughs in one field based on previously known results in another? (#40)
  + ROI on science/{$,time}invested (#36)
    - more discoveries! (#37)
  + More accessible computation → more citizen science (#38)
  + benefits a wide range of community (not applications that leverage supercomputing) ... (#39)
  + Less duplication of efforts through transferable tools, skills, and knowledge (#41)
  + Replacing costly, dangerous, or otherwise unsavory experimentation with computer simulation (#43)
    - Digital twins accelerating science/experiments (#42)
* 4. Who needs to be involved: science domains and research communities
  + Comments
  + This needs to not just be an ASCR program, but it needs to span all of DOE, including efforts in the domain sciences. (#29)
    - mechanisms for extending beyond DOE as well (#47)
  + International collaborations (large scale societal problems from developing nations related to poverty, climate change, flooding etc) (#45)
    - This is also true for just international scientific collaborations (#48)
  + Representatives from congress (bipartisan support) (#49)
  + Computer science (AI, compilers, programming languages, etc.), computer engineering (architecture, hardware, etc.), software engineering, computational scientists, IT, cyber security -- as well as domain scientists from all disciplines (#50)
  + University, industry partnerships (#52)
  + DOE, or whoever winds up managing this, should try to avoid pre-selecting a lead institution that wins big at the expense of all the other players. Success will likely require many institutions to contribute equally. (#53)
* 46. What staff and management communities and what kind of structure needs to be involved.
  + Comments
  + Funding structures that \*mix\* different offices of science and beyond (#51)
  + Many autonomous groups working on projects, encouraged to reach out and work with other groups. One larger overseeing group which facilitates inter-operation between groups. (I.e. don't get too stuck in hierarchies/management, allow groups to find best solutions themselves) (#54)
    - more than facilitate, actively prevent barriers from arising between groups (#55)
  + Central structure with administrative/business/project management staff, but researchers that are involved are distributed across many other organizations? (#56)
    - role of management is to facilitate, not to manage/dictate what organizations/groups work on (#57)

Appendix

Live chat

**The following messages were exchanged via Live chat.**

* Nov 2, 2020, 17:30 UTC
  + My Breakout 1 link isn't doing anything. (Scott Pakin (LANL) | Nov 2, 2020, 17:30 UTC)
  + I'm going to try exiting and going back in. (Scott Pakin (LANL) | Nov 2, 2020, 17:32 UTC)
  + Ah, that worked. (Scott Pakin | Nov 2, 2020, 17:32 UTC)
  + I'll be a few mins late for the second session. I'll join once my 2pm meeting finishes. (Neil Thompson | Nov 2, 2020, 19:32 UTC)
  + Thanks for letting us know (Yasaman Ghadar | Nov 2, 2020, 19:43 UTC)