Towards Scalable Execution Across Multiple XSEDE Resources

Shantenu Jha

Abstract

Multi-site execution (MSE) is the ability to utilize multiple resources collectively and concurrently. There is lack of understanding of the concepts, of the requirements, and the challenges in implementing MSE. This limits the effective and efficient utilization of distributed cyberinfrastructure as well as throttles innovation in application usage. This white paper will present a fresh perspective on MSE, including a roadmap to help alleviate these problems. We propose a simple yet powerful conceptual view of MSE, survey the landscape of MSE, and discuss two illustrative examples of improvements in scientific applications as well as production distributed cyberinfastructure, including but not limited to XSEDE and OSG

1 Introduction and Context

The cyberinfrastructure community has been providing production distributed cyberinfrastructure (DCI) for more than a decade and a half to support NSF's science & engineering efforts. We are still learning how to architect large-scale production DCI [1, 2]. This is evidenced by missing design principles and an absence of scalable architecture for DCI. Signatures of these gaps can be found in the following three observations:

- Distributed applications are characterized by gluing them to specific platforms. There is lack of a minimally complete set of abstractions for application development, deployment and execution. This results in applications that are brittle as a consequence of being dependent upon local and point solutions.
- NSF's existing portfolio of production DCI – consisting primarily of XSEDE and OSG, are incompatible, inconsistent and not interoperable. Minor differences in the starting points of distinct DCI result in very different end points. Such chaotic sensitivity to initial conditions reiterates the lack of design principles for DCI. Furthermore, there are many different external and internal interfaces, deployment and execution constraints and requirements; this complexity reinforces the fundamental challenges inherent in developing distributed applications.
- There exists a complete inability to reason about spatial-temporal execution of distributed workloads and answer some basic questions about their execution. For example: Which resources should a workload use? What time-to-completion should a workload expect? How do specific execution decisions influence the performance? Models to answer such questions, if any, are customized to specific application scenarios and infrastructure, and are not general purpose, nor do they support end-to-end reasoning.

Against this backdrop, it is no surprise that there are missing methods, inadequate tools, and insufficient conceptual understanding of the concurrent and collective utilization of resources. Although at some level this is merely a reflection of the broader context of DCI, it is somewhat ironical, as the raison d'etre of DCI such as XSEDE and OSG is to ensure that the collective capabilities and services provided "as a whole are greater than the sum of the parts".

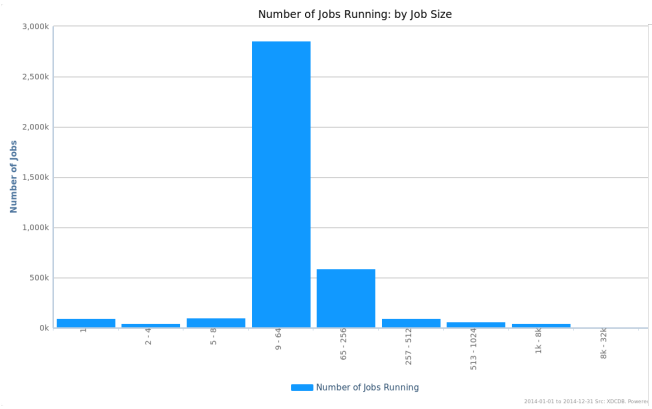

Figure 1: The distribution of job sizes submitted to XSEDE in 2014. We have investigated the distribution for Stampede and the distribution is essentially similar to that of XSEDE as a whole.

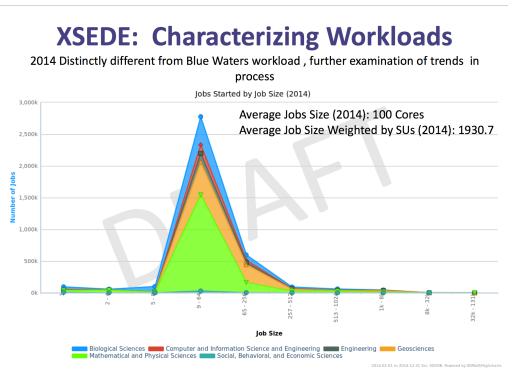

Figure 2: The distribution of job sizes submitted to XSEDE decomposed by domains (source: Irene Qualters, BDEC Barcelona 2015); interestingly the distribution is similar across the primary domains.

In this white paper, the principled usage of MSE as a necessary and existential attribute for DCI is reinforced by the potential of its practical impact. We now discuss briefly three such practical advantages.

Novel Usage Modes: The collective utilization of resource, should enable a research scientist to harness the power of multiple resources with essentially the same effort as required to use any given individual resource as well as reach larger scales than otherwise possible. MSE is a prerequisite in supporting novel distributed usage modes; novel and scalable usage of DCI in turn can lead to scientific advances. For example if workloads can be distributed they can use specialized platforms, data collections or be executed using user-defined objectives as opposed to being executed using single resources.

Reduced time-to-completion: The ability to efficiently place workloads or seamlessly re-distribute workloads, e.g. from heavily loaded machines onto less loaded machines, will result in a reduction in the time-to-completion (TTC) as opposed to a "best effort" resource response times. For example, the seamless redistribution of jobs from an overloaded Stampede to under loaded Comet without loss of performance will result in lowered TTC.

Improved System Utilization: Workload distribution using MSE has the potential to improve resource utilization. For example, distributed and multi-resource workload balancing enables meeting increased demand whilst improving utilization. In general better matching of workloads to heterogeneous resource types has immense potential.

Scope for MSE: What fraction of XSEDE applications can in principle take advantage of the ability to scalably execute across multiple federated resources? A simple analysis points to a potentially large set of applications based upon size and type. Figure 1. plots the distribution of (aggregated) job sizes on XSEDE in 2014. It shows that 80% of all jobs submitted to XSEDE (i.e., all resources that make up XSEDE) were between 8 and 64 cores. Jobs in this size range are essentially resource agnostic and thereby platforms are fungible. Thus, considered just by their size, up to 80% of all jobs submitted to XSEDE are suitable to be executed in a manner that does not bind them to any specific resource. Although solutions like back-filling may not be possible for all 80%, we know that a significant fraction of jobs submitted to XSEDE are independent jobs, and thus can exploit advantages arising from MSE. The approximately 350K single node and essentially independent jobs submitted to XSEDE annually [3, 4] are particularly suitable [5]. Whereas size does matter, we understand it is not the only consideration in determining suitability for MSE. Applications that are comprised of multiple uncoupled tasks or coupled components are also candidates provided the details of distribution can be managed or abstracted away [6].

2 MSE: Core Concepts

Multisite Execution is often presented as a single monolithic problem, but like most other grand challenges and visions, it is in fact a combination of many smaller, conceptually independent but functionally correlated challenges. Thus there is a need to "normalize" and "formalize" MSE, until which not only will our ability to scalably federate and execute tasks seamlessly over XSEDE remain limited, so will our ability to make conceptual progress.

Our analysis suggests that there are three distinct core concepts underlying MSE: (i) the construction of a resource overlay (i.e., resource federation) which can be used to execute the workload, (ii) the need to coordinate workloads and map them onto federated resources, and (iii) enable the delivery and execution of workloads as tasks and associated data, and handle execution details such as stage-in/out of data, resolve dependencies, and monitor and control the execution on the federated resource.

We discuss some of the technical responsibilities and requirements of the logical components corresponding to these core concepts.

Resource Federation: There is a need to determine which resources can be used, which resources should be used and how to do so. In some simple instances the resources are specified but the challenge of presenting the resources via a common interface remains. The resource federation component should be responsible for: determining resource availability, state and capacity; creating the federated pool of resources and addressing spatio-temporal fluctuations in resource availability.

Workload Mapping and Management: Given the broad range of application types, heterogeneity of resources and application performance constraints, the workload management and mapping component must address the following: Manage workloads over resources whose state and properties are time-dependent, as well as provide a general purpose mapping of workloads to (federated) resources.

Task Execution: The semantics of task execution must remain independent of the resource specifics. Thus any task execution system must provide the following capabilities: Transform and deliver the workload to multiple distinct machines, whilst managing heterogeneity across resources that influence execution details such as stage-in/out of data, resolve dependencies, and monitor and control the execution on the federated resource.

3 MSE: Status quo

In order to highlight the gap of what is possible in principle versus what exists in practise, we discuss the state of MSE at three levels: (i) Individual, stand-alone applications and projects, (ii) Tools and (iii) System-level capabilities.

MSE at the level of individual applications: In spite of the theoretical advantages of MSE, the number of applications that actually employ MSE is very small. Specific applications have been able to use multiple resources [7] after heroic efforts, but not by using general purpose, reusable or extensible approaches.

MSE at the Tool Level: Although some large projects have been able to utilize MSE, individual PIs still struggle. The lack of democratic MSE approaches and solutions point to several underlying problems, including problems with the ecosystem of tools, which in turn point to the lack of abstractions and building blocks for MSE. For example, successful as gateways have been in lowering the access barrier to individual resources, the bulk of gateways do not support MSE. If gateways were built upon well-defined building blocks for MSE, each gateway could focus on the application specific part, as opposed to developing software components which are either redundant or provide a functionally incomplete set of MSE capabilities.

The situation is qualitatively somewhat similar to workflow tools: there are many workflow tools but well-defined and reusable resource federation and task execution components do not exist. Consequently, most workflow systems do not focus on innovation at the programmatic/application level, as they are often trapped in managing the complexity of resource aggregation and task-execution in distributed environments.

There are a plethora of tools that can support MSE in principle. These tools however, are point-solutions, often non-extensible, and often fail to provide conceptual and functional separation between the many functional components that MSE entails. Condor which is commonly used by applications and higher-level tools represents the state-of-affairs: it has become a kitchen-sink for "distribution" "workload management" "resource management" "infrastructure access", inter alia, and thus violates functional layers as well breaks abstractions. Condor fails to cleanly and clearly separate the many integrated but functionally distinct components that underpin MSE; not surprisingly that " propagates" upwards to tools using it.

MSE at the System Level (on XSEDE): There are no system-level services on XSEDE that implicitly or explicitly provide MSE capabilites. Load balancing across resources and workload redistribution is essentially manual and it is the responsibility of individual projects and users. In fact, given the commonality of workload and resource management requirements across projects (for example, ATLAS, CMS and ALICE high-energy experiments), they should in principle all use similar approaches, if not the same tool. Not surprisingly, as projects try to break free of their traditional execution modes, they face both fundamental and practical challenges. Similarly projects such as iPlant have their own customized approaches and tools, which do not delineate the core principles of scalable and extensible MSE.

4 The Road Ahead: The Promise of MSE

MSE is currently characterized by resource and workload specific approaches, as well as a lack of distinction between implementation specific issues from the core conceptual issues. There are several reasons however, that motivate a fresh and comprehensive perspective to MSE. First and foremost is the potential for bigger and better science; in addition to which there are system-level advantages. By necessity, we briefly discuss two simple scenarios.

Unified OSG and XSEDE MSE: The core concepts ($\S$2) are common to both XSEDE and OSG. They have the potential to provide a common vocabulary and yield common design principles, something that is currently missing. Advances in conceptual clarity and uniformity of the principles of MSE will not guarantee or in of itself deliver interoperability between OSG and XSEDE. It will however, be an important step towards the general-purpose and scalable interoperation between XSEDE and OSG, as opposed to the plethora of ongoing efforts, each looking for point solution (c.f. ExTENCI project), interoperability at the lowest common denominator, and for a quick fix solutions, without an appreciation for the broader context or concerted efforts at reuse.

Improved System Responses: A large fraction of the $10^4 - 10^5$ jobs submitted every day to XSEDE can in principle be assigned to any of the O(10) resources. However, currently a very large fraction of all jobs are bound to a specific XSEDE machine, thus it is no surprise that there are significant queue waiting times and machine-to-machine fluctuation on XSEDE. For example, queue waiting times on a given machine on XSEDE vary by a factor of two over a window of 24 hours, as do instantaneous utilization/load factors [3].

When averaged over sufficiently large time-scales and number of resources, there inevitably exists some spare capacity on individual resources and XSEDE as a whole. Imperfect load-balancing across XSEDE as a collective, is in essence a gross under-utilization of resources.

It would help remove this bottleneck somewhat, if jobs were to be assigned to an XSEDE resource based upon some measure of relative utilization/load, as opposed to being localized to a specific resource. This will not only lower average queue waiting time, it would improve the utilization of individual XSEDE machines and thus essentially improve overall throughput of the XSEDE.

Towards an XSEDE Execution Management Service It follows that an XSEDE-wide multi-site execution management service (EMS) that adhered to well-understood principles and encapsulated the three functional requirements (resource federation, workload mapping and task execution) would be very effective for system utilization. Given the broad range of resources in NSF's arsenal and application types – from the long-tail of science to big projects – that the NSF's DCI must support, any solution will need to move beyond building customized solutions that support only a specific workload type, distribution and usage mode.

The need is to provide base capabilities, upon which it is possible to build a set of common and

extensible set of higher-level services and interoperable capabilities [8]. In addition to working across resource (compute, data and network) types and for diverse application types, any EMS must be extensible to different mechanisms of federation and workload mapping, and across different scales, viz., Works for O(10) tasks as well as O(10000) tasks; works for O(10) sites as well as O(100) sides.

There are a plethora of issues that must be addressed before the above requirements can be met: What is the appropriate scale of resource federation and workload aggregation? Given Figure 2, should an EMS be domain-specific, or XSEDE wide? Given spatio-temporal fluctuations of resources, should all XSEDE (and campus) resources be federated into one large virtual resource, or is there an optimal granularity? How much control should be established over the federation vs adapt to the federation? How does the degree of heterogeneity of resources and applications influence the answers? How to respond to changes in resource availability, load and changes in workload requirements? How does the properties and type of workload influence the outcome?

Eventually advances in answering these questions should yield formal models, design principles and scalable architectures for an XSEDE wide execution management service. Without well-defined models, methods and software components that implement core MSE functionality, they will have limited impact.

Conclusion

In this white paper we have identified the potential impact of MSE but have also identified some of the existing gaps. The current inability to utilize MSE is one important reason preventing applications from scaling to the requisite level and using the computational capabilities that their science demands. The inability to utilize multiple resources collectively in addition to being a technical challenge also poses a resource utilization and sustainability problem. We believe this unsatisfactory state of affairs needs addressing. We advocate a systems science approach to MSE, one that identifies core conceptual constituents, derives functional building blocks and provides a framework to reason about implementation and other design decisions. This white paper provides a timely reminder that with advances in MSE, DCI such as XSEDE will be able to support the collective utilization of their resources for a broad range of applications and usage modes.

Acknowledgement: The author would like to thank Prof. Kaushik De (UTA) for helpful comments and insight into the requirements of the ATLAS project. The author would like to thank members of the RADICAL research group (http://radical.rutgers.edu) for comments and discussion.

References

[1] The Science of Cyberinfrastructure: Research, Experience, Applications and Models (SCREAM) Workshop
`https://sites.google.com/site/scream15workshop/`.

[2] Shantenu Jha, Murray Cole, Daniel S. Katz, Manish Parashar, Omer Rana, and Jon Weissman. Distributed Computing Practice for Large-Scale Science & Engineering Applications. *Computing and Concurrency: Practise and Experience*, 2012. Volume 25, Issue 11, pages 15591585, 10 August 2013 (doi: 10.1002/cpe.28 97).

[3] XDMOD Portal. XD Metrics on Demand.
`https://xdmod.ccr.buffalo.edu/#tg_usage:group_by_Jobs_none`.

[4] Using XDMoD to facilitate XSEDE operations, planning and analysis XSEDE '13 Proceedings of the Conference on Extreme Science and Engineering Discovery Environment: Gateway to Discovery, doi 10.1145/2484762.2484763.

[5] There are three associated points worth discussion: (i) There is no rigorous definition/cut-off after which jobs are best submitted to a single, specific resource. After a certain size, a workload is less suitable for resource-agnostic execution as they require more work to exploit architectural features in order to scale, or conversely they have undergone such resource-specific optimization. (ii) It is worth mentioning that many jobs on XSEDE employ community codes, which are already ported across resources, so

this addresses some of the possible qualitative and socio-technical barriers. (iii) An earlier independent analysis in 2012 in by XDMoD team estimated the percentage of XSEDE jobs in this range to be closer to 70%. We will not analyze whether this is a trend or just a fluctuation, nor will we conjecture reasons for this shift from 2012 to 2014, but focus on the consistent conclusion that there is a large percentage of jobs that can be submitted to any resource.

[6] These multi-component applications could be part of a larger workflow, or aggregated community jobs. Needless to say the technical challenges are greater than uncoupled independent tasks, but given the increasing proportion of applications that exhibit heterogeneous, diverse workloads and multi-component workloads (e.g., multi-components representing multi-physics applications such as coupled-climate models and multi-scale simulations, the search for novel execution paradigms and innovative solutions are well worth it.

[7] G. Allen, T. Dramlitsch, I. Foster, N. Karonis, M. Ripeanu, E. Seidel, and B. Toonen. Supporting efficient execution in heterogeneous distributed computing environments with cactus and globus. In *Proceedings of Supercomputing 2001*, Denver, USA, 2001.

[8] As more projects come online (e.g., LSST and other observation based experiments) and integrate with XSEDE, their requirements need to be addressed in a general purpose and extensible way, so as to prevent multiple inconsistent and redundant efforts. Whereas it might just be technically possible for large-projects to develop their own customized software infrastructure for MSE, however, for all major large projects to do so, is unproductive, and ultimately not sustainable. Furthermore, that is rarely an option for the long-tail of science, where individual domain scientist “just need tools to get the science done”.