METRICS

Ease of Programming	How much in-depth knowledge of processor architecture is required for a developer before starting to convert the code to GPU?
Performance	How much effort is required to reach desirable performance on a particular architecture.
Portability	To what extent does the same code run on multiple architecture? What provisions are provided by programming approach to target different platforms?

Support	The overall ecosystem and support by the community.
	    Which all compilers implement the standard?
	    Which all languages are supported?
	    Which all applications make use it?
	    How easy or difficult it is to profile/debug the application?

MATRIX

Ease of Programming	High: Minimal architecture specific knowledge needed	Intermediate: Minimal changes expected in code design. Using these along with architecture knowledge helps in better performance	Low: In-Depth GPU architecture knowledge must
Performance	Depends: Based on the complexity/type of application the performance may vary	High: Exposes methods to get good performance. These methods are integral part of design and exposed to programmer at various granularities	Best: Full control to developers to control parallelism and memory access
Portability	Integral: Part of the key objective	Limited: Works only on specific platform	None: Architecture specific
Support	Established: Proven over years and support by multiple vendors for GPU	Emerging: Gaining traction by multiple vendors for GPU	

MATRIX

	OpenACC	OpenMP	stdpar/do-concurrent	Kokkos	CUDA Languages
Ease	High	High	High	Intermediate	Low
Performance	Depends	Depends	Depends	High	Best
Portability	Integral	Integral	Integral	Integral	None
Support	Established	Emerging	Emerging	Established	Established

OpenACC
More Science, Less Programming

OPEN
HACKATHONS

Acknowledgment

Learn more at

WWW.OPENHACKATHONS.ORG

OpenACC
More Science, Less Programming

OPEN HACKATHONS