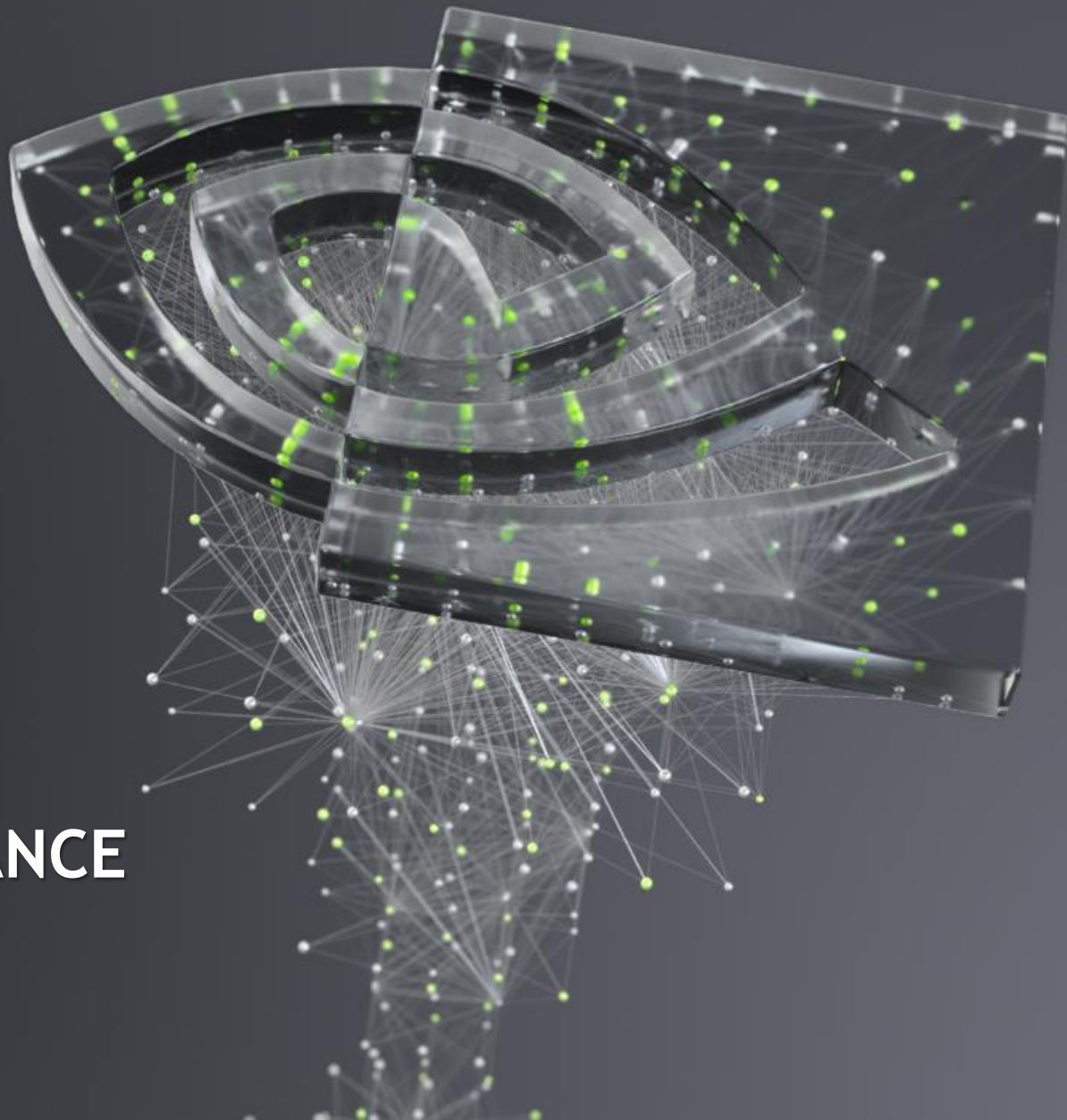




# VASP 6.2

## ACCELERATED PERFORMANCE

July 2021





# AGENDA

---

Introduction to VASP

---

Supported Accelerated features in VASP 6.2

---

Performance of VASP on NVIDIA

---

Operational benefits of NVIDIA technology

---

# INTRODUCTION TO VASP

## Scientific Background

Most widely used GPU-accelerated software for electronic structure of solids, surfaces, and interfaces

### Generates

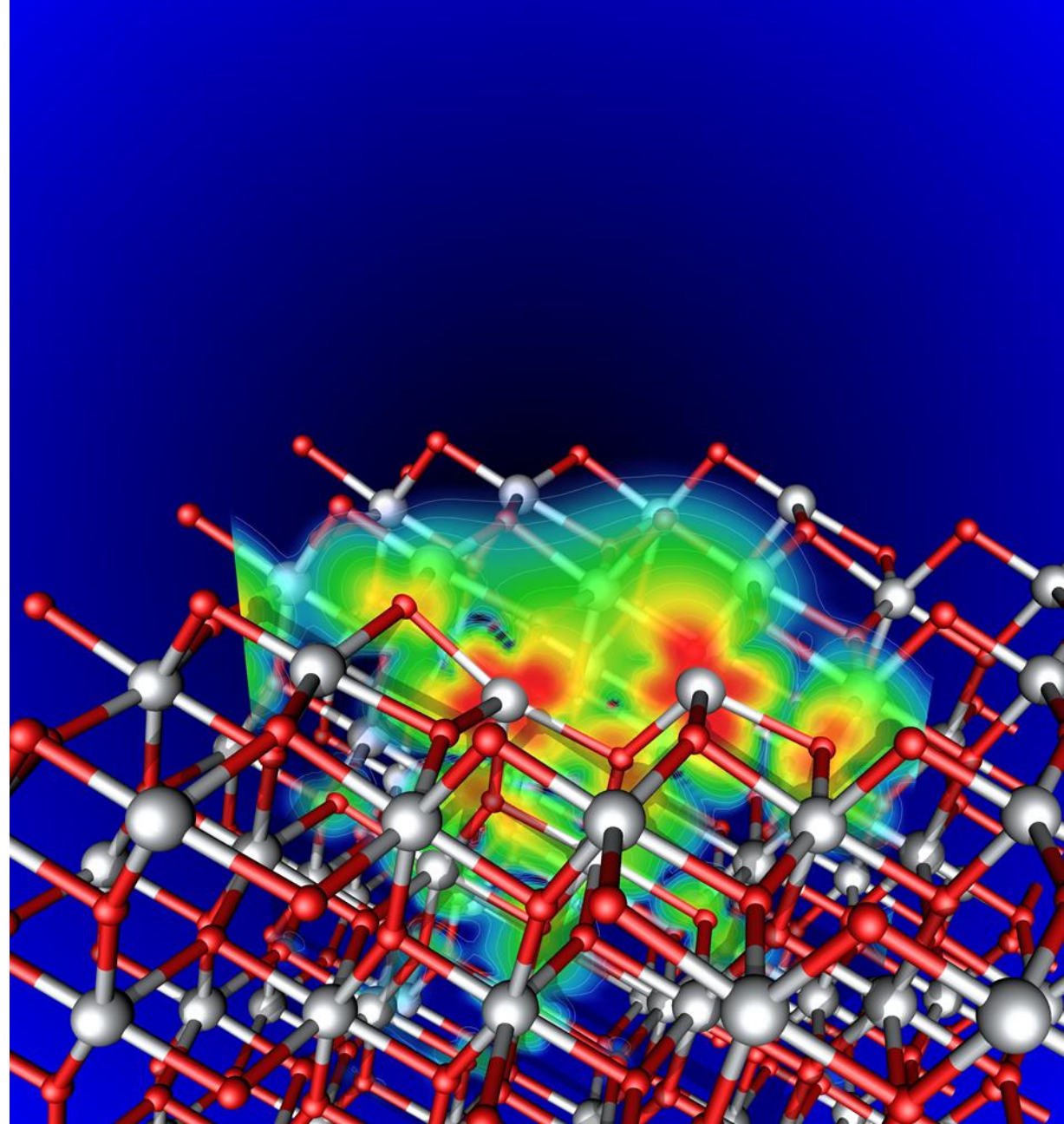
- Chemical and physical properties
- Reactions paths

### Capabilities

- First principles scaled to 1000s of atoms
- Materials and properties - liquids, crystals, magnetism, semiconductors/insulators, surfaces, catalysts
- Solves many-body Schrödinger equation

### Quantum-mechanical methods and solvers

- Density Functional Theory (DFT)
- Plane-wave based framework
- New implementations for hybrid DFT (HF exact exchange)



# VASP SOFTWARE ORIGINS

## Key facts

Developed by Kresse group at the University of Vienna and VASP Software GmbH

Development began >25 years ago

460K lines of Fortran code

MPI parallel, OpenMP recently added for multicore

GPU acceleration efforts started prior to 2011 with CUDA C

## Computational characteristics

Many small Fast-Fourier-Transformations  $\sim 100^3$

All-to-all communications

Matrix operations

- Matrix-Matrix multiplications
- Matrix-Vector multiplications
- Diagonalizations

Custom kernels



# AGENDA

---

Introduction to VASP

---

Supported Accelerated features in VASP 6.2

---

Performance of VASP on NVIDIA

---

Operational benefits of NVIDIA technology

---

# FEATURES AVAILABLE AND ACCELERATED IN VASP 6.2

## LEVELS OF THEORY

Standard DFT (incl. meta-GGA, vdW-DFT)  
Hybrid DFT (double buffered)  
Cubic-scaling RPA (ACFDT, GW)  
Bethe-Salpeter Equations (BSE)

## SOLVERS / MAIN ALGORITHM

Davidson (+Adaptively Compressed Exch.)  
RMM-DIIS  
Davidson+RMM-DIIS  
Direct optimizers (Damped, All)  
Linear response

## PROJECTION SCHEME

Real space  
Reciprocal space

## EXECUTABLE FLAVORS

Standard variant  
Gamma-point simplification variant  
Non-collinear spin variant

- Existing acceleration
- New acceleration
- Acceleration work in progress
- On acceleration roadmap

# FEATURES AVAILABLE AND ACCELERATED IN VASP 6.1

## LEVELS OF THEORY

Standard DFT

Hybrid DFT (double buffered)

Cubic-scaling RPA (ACFDT, GW)

Bethe-Salpeter Equations (BSE)

## SOLVERS / MAIN ALGORITHM

Davidson (+Adaptively Compressed Exch.)

RMM-DIIS

Davidson+RMM-DIIS

Direct optimizers (Damped, All)

Linear response

## PROJECTION SCHEME

Real space

Reciprocal space

## EXECUTABLE FLAVORS

Standard variant

Gamma-point simplification variant

Non-collinear spin variant

- Existing acceleration
- New acceleration
- Acceleration work in progress
- On acceleration roadmap

# FEATURES AVAILABLE AND ACCELERATED FROM VASP 5

## LEVELS OF THEORY

Standard DFT

Hybrid DFT

RPA (ACFDT, GW)

Bethe-Salpeter Equations (BSE)

...

## SOLVERS / MAIN ALGORITHM

Davidson

RMM-DIIS

Davidson+RMM-DIIS

Direct optimizers (Damped, All)

Linear response

## PROJECTION SCHEME

Real space

Reciprocal space

## EXECUTABLE FLAVORS

Standard variant

Gamma-point simplification variant

Non-collinear spin variant

- Existing acceleration
- New acceleration
- Acceleration work in progress
- On acceleration roadmap





# AGENDA

---

Introduction to VASP

---

Supported Accelerated features in VASP 6.2

---

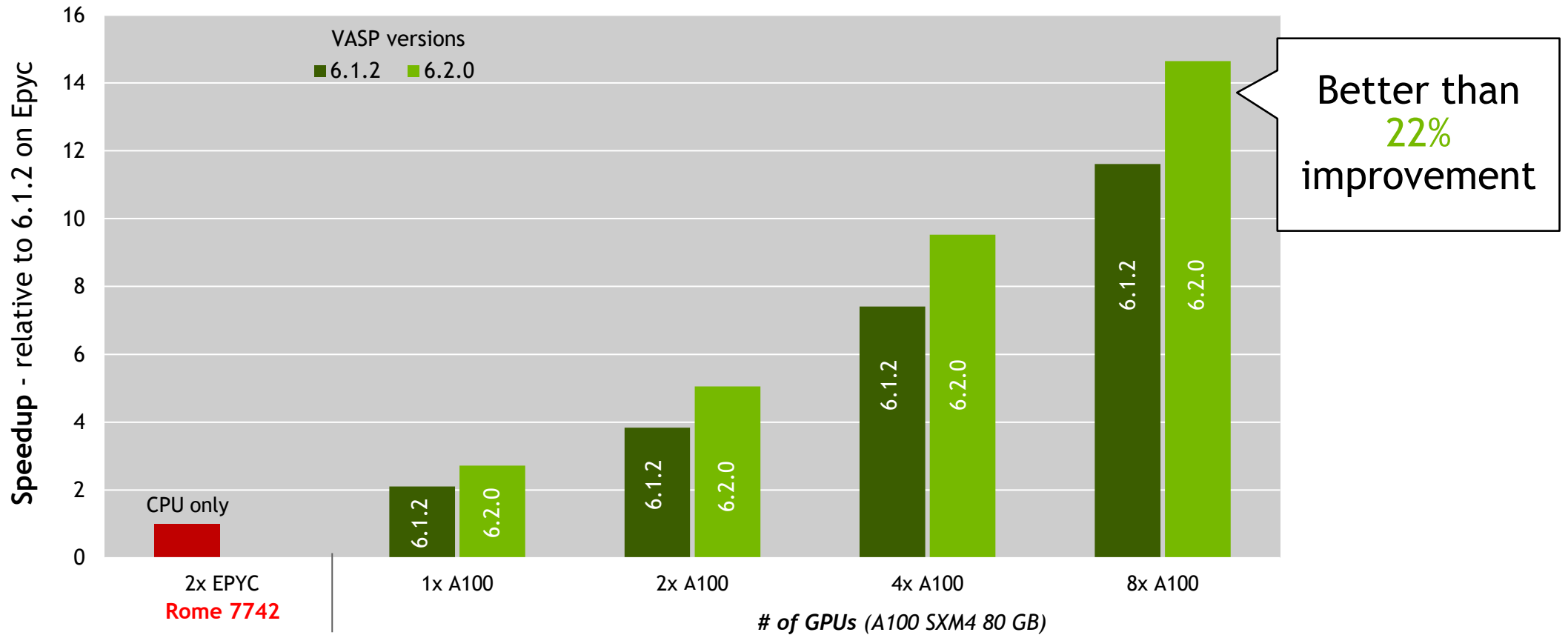
Performance of VASP on NVIDIA

---

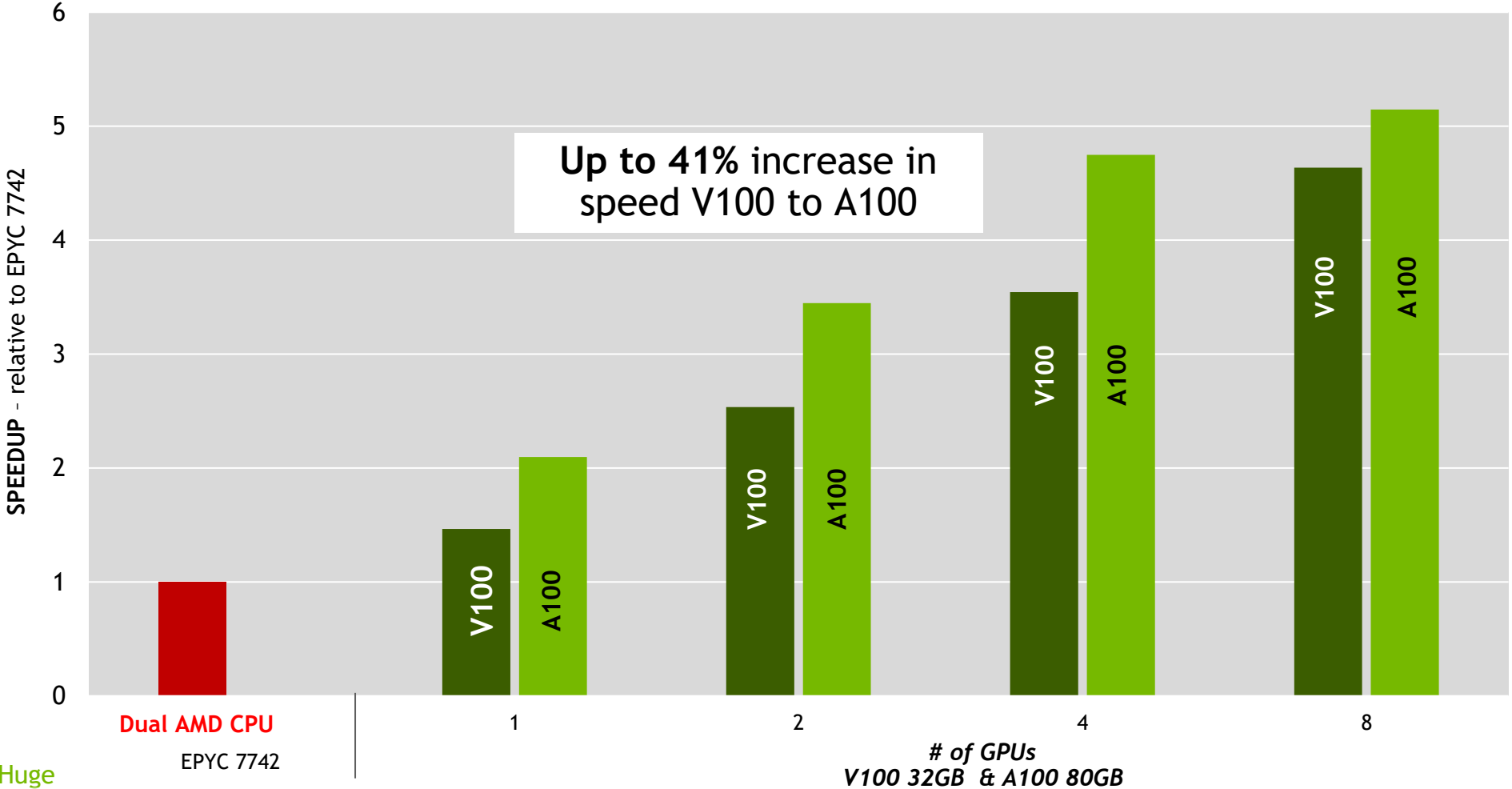
Operational benefits of NVIDIA technology

---

# VASP VERSION UPDATES BRING NEW ACCELERATION



# NEW NVIDIA GPU PLATFORMS - ADDITIONAL ACCELERATION



Dataset: Si-Huge  
VASP 6.1.2

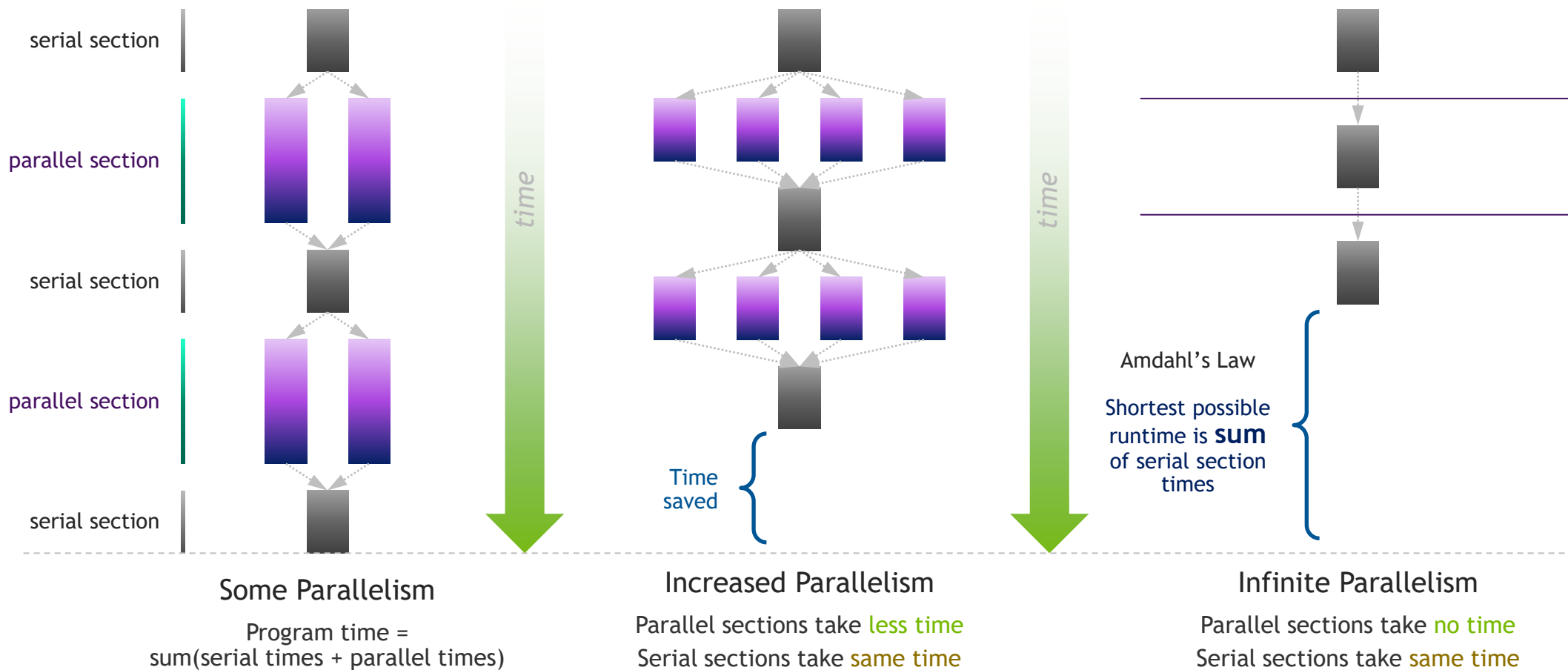


# VASP - 6.2.0

CPU-only: 2xEPYC 7742 GPUs: A100-SXM4-80GB with HPC SDK 21.2 and CUDA 11.0

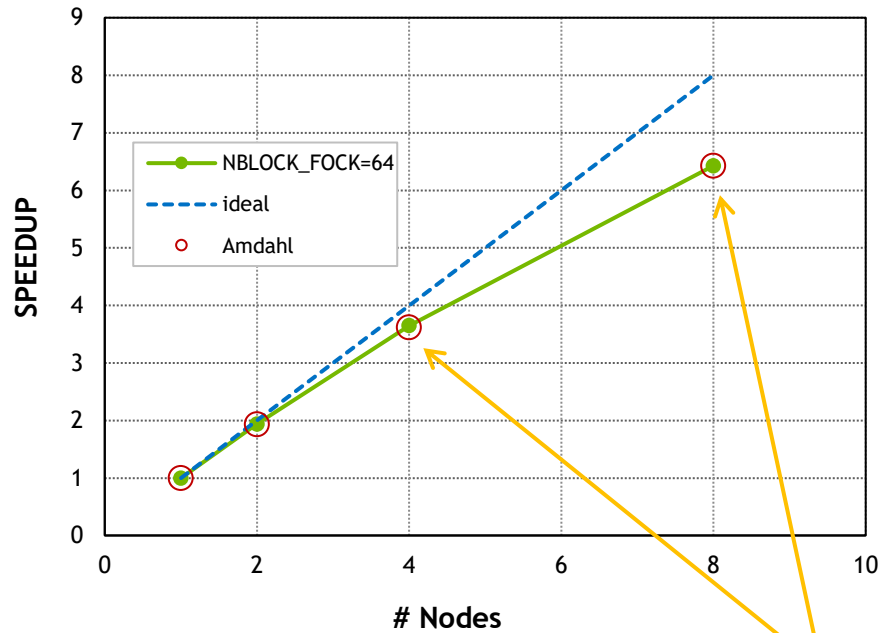


# AMDAHL'S LAW

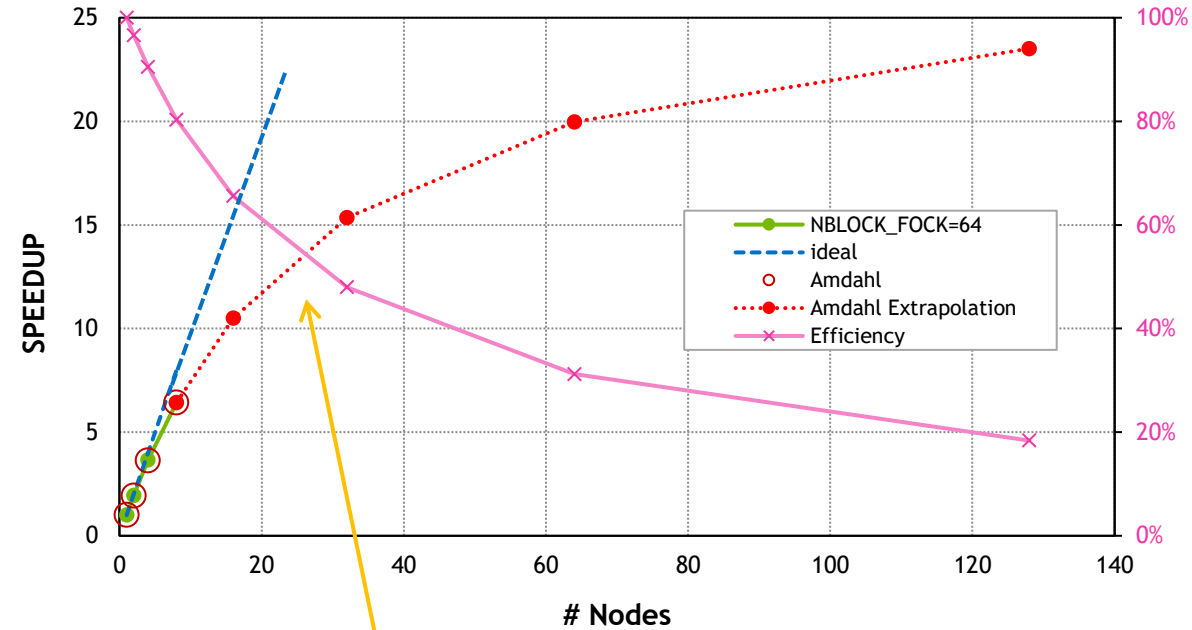


# MULTI NODE VASP - SCALING EXAMPLE

8 V100 GPUs nodes connected with HDR Infiniband



With an Amdahl's law numerical fit the approximation is quite good



50% Scaling efficiency occurs at 30 nodes in this example



# AGENDA

---

Introduction to VASP

---

Supported Accelerated features in VASP 6.2

---

Performance of VASP on NVIDIA

---

Operational benefits of NVIDIA technology

---

# WHY VASP DEVELOPERS CHOSE OPENACC



*“ For VASP, OpenACC is the way forward for GPU acceleration. Performance is similar and in some cases better than CUDA C, and OpenACC dramatically decreases GPU development and maintenance efforts.”*

Prof. Georg Kresse

CEO of VASP Software GmbH

Computational Materials Physics  
University of Vienna



# VASP 6 WITH OPENACC

## Hardware requirements and recommendations

Works on all architectures supported by NVIDIA: x86, POWER and ARM

Ideally all GPUs connect with 16 PCIe lanes to the CPUs, otherwise use PCIe switches to share lanes with NICs

Best performance on NVIDIA GPUs with strong double precision (FP64) capabilities on A100, A30 is also an option. Volta generation V100 continues to provide excellent performance.

NVLink GPU-GPU-interconnects speed-up AllToAll communication

Dense GPU nodes preferred for throughput, fast network like Mellanox Infiniband is essential

# VASP 6 WITH OPENACC

## Software requirements and recommendations

[NVIDIA HPC SDK 21.5](#), no cost and includes requirements

- OpenACC compiler (formerly PGI)
- NVIDIA CUDA Toolkit and Libraries: cuBLAS, cuFFT, cuSOLVER and NCCL
- CUDA-aware MPI (OpenMPI 3.1.5 without UCX recommended; otherwise use UCX  $\geq 1.9$ )

CPU math libraries: FFTW (compile with GCC, don't use OpenMP support), OpenBLAS and ScaLAPACK

# VASP 6.2 WITH OPENACC

## How to compile

HPC SDK brings all dependencies besides `FFTW`, so you only need to adapt this variable in `makefile.include` to match the path on your system, or export them as an environment variable, e.g.:

```
export FFTW=/opt/fftw-3.3.9
```

It is recommended to build on the target system, otherwise add the appropriate `-tp` flag to the `FC`, `FCL`, `FC_LIB`, `CC_LIB` and `CXX_PARS` lines

Build the binaries accelerated using OpenACC:

```
make std gam ncl DEPS=1 -j 10
```

# VASP 6 WITH OPENACC

## How to run the accelerated version

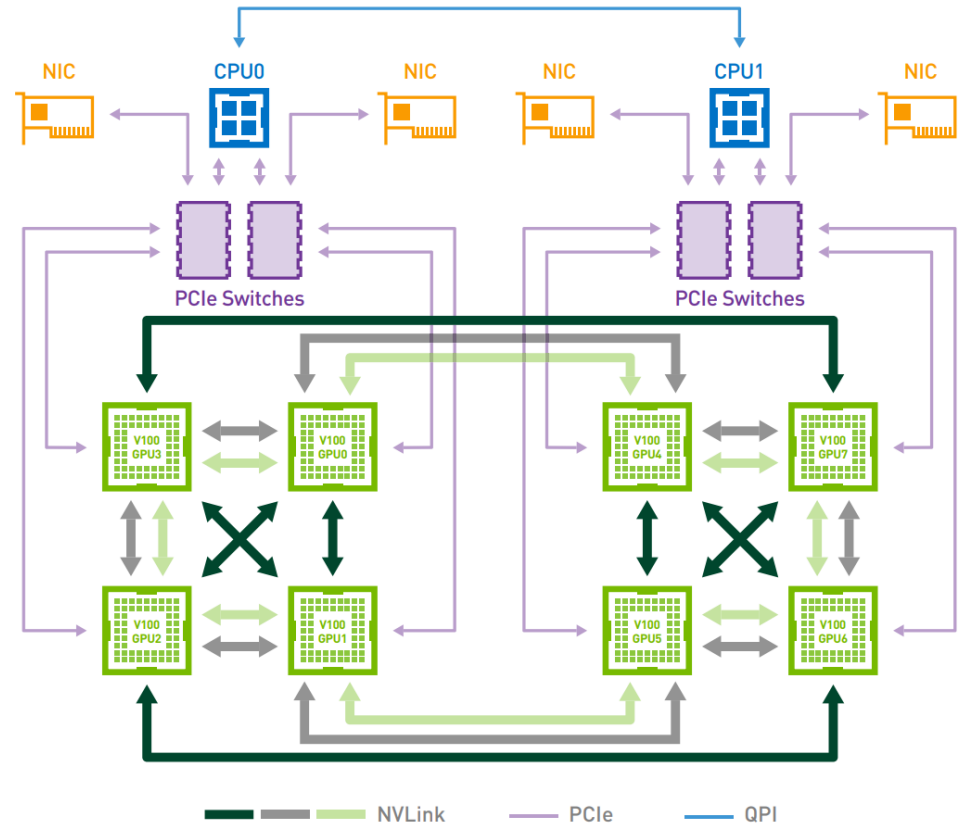
Run with 1 MPI rank per GPU  
(requirement by NCCL library; don't use MPS as with the CUDA-C-port anymore)

Restrict libraries (like OpenBLAS or FFTW) to run with 1 thread per process only

VASP will select the GPUs automatically and use them in sequential order: Rank 0 → GPU 0, Rank 1 → GPU 1, ...

Bind your processes to the CPU sockets with correct affinities to the GPUs and NICs. In doubt check with

```
$ nvidia-smi topo -m
```



# VASP 6 WITH OPENACC

## Binding your processes with correct affinities

Use a script like the following and run with `mpirun -n 8 runscript.sh vasp_std`

Example `runscript.sh` for DGX1:

```
#!/usr/bin/env bash
export UCX_RNDV_THRESH=1024
export UCX_MEMTYPE_CACHE=n
export OMP_NUM_THREADS=1
NICS=(mlx5_0 mlx5_0 mlx5_1 mlx5_1 mlx5_2 mlx5_2 mlx5_3 mlx5_3)
CPUS=(0 0 0 0 1 1 1 1)
lrank=$OMPI_COMM_WORLD_LOCAL_RANK
export UCX_NET_DEVICES=${NICS[$lrank]}:1
export OMPI_MCA_btl_openib_if_include=${NICS[$lrank]}
numactl --cpunodebind=${CPUS[$lrank]} --membind=${CPUS[$lrank]} $@
```

# VASP 6 WITH OPENACC

## Tune your VASP jobs

Use `vasp_gam` binary when possible! Saves memory and faster execution

INCAR: Remove `NPAR` and set `NCORE=1`: VASP 6.1.2 will do this for you internally, but better be safe.

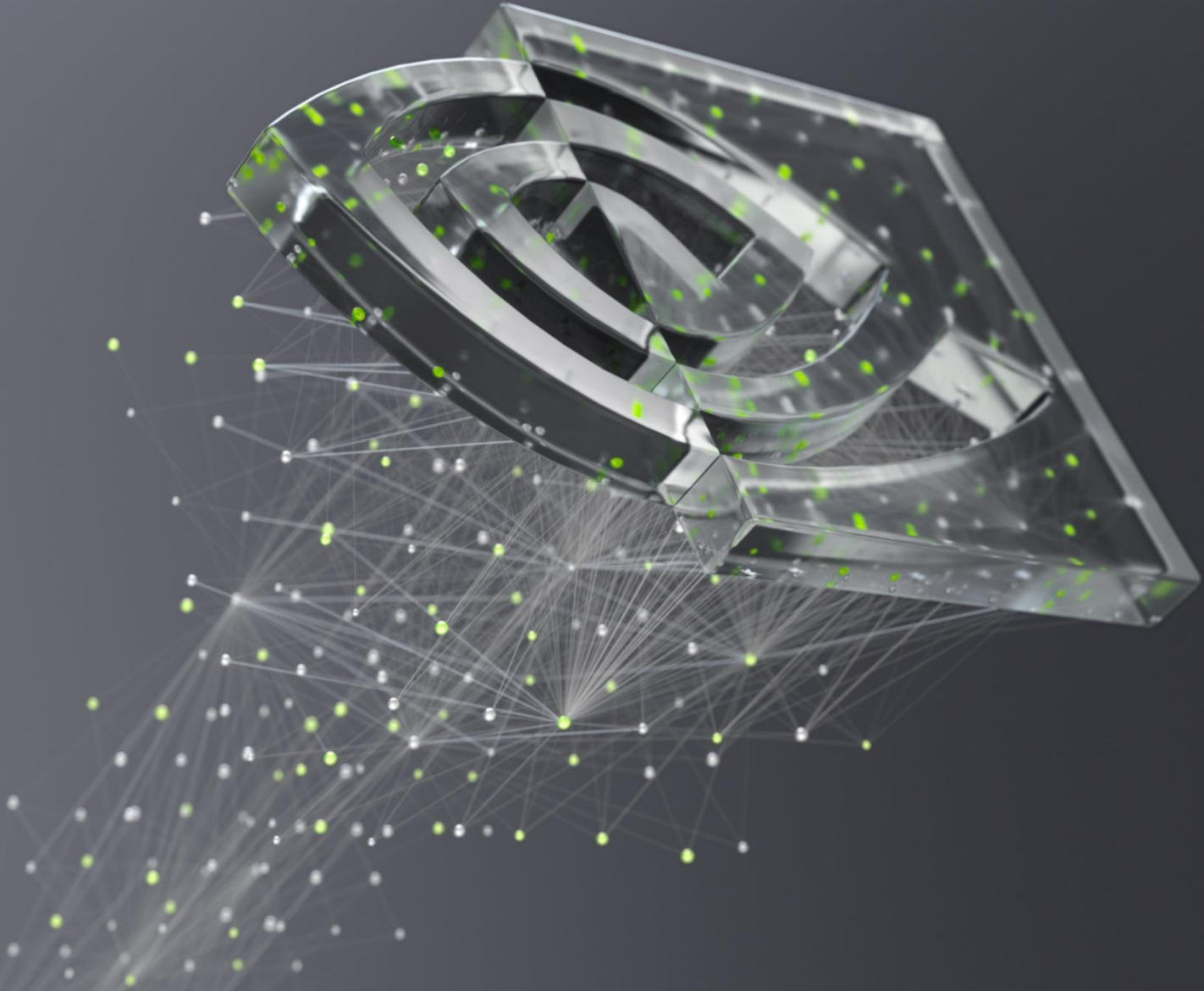
INCAR: For `vasp_std` and `vasp_ncl` jobs, set `KPAR`: Use a value that evenly divides the number of k-points (`grep NKPTS OUTCAR`) by the number of GPUs. The higher the better. Much improved performance for increased memory usage.

INCAR: For standard and hybrid DFT jobs tune `NSIM` parameter. Test powers of 2 until it uses too much memory or performance stops improving.

INCAR: For hybrid DFT jobs, tune `NBLOCK_FOCK` parameter. Use a value that evenly divides the number of bands/orbitals (`grep NBANDS OUTCAR`) by the number of GPUs. As a rule of thumb, the higher the better.

# VASP RECOMMENDED USAGE PLATFORM

Motherboard and CPU	Single or Dual-socket CPU
System memory	$\geq 32\text{GB}$
NVIDIA GPU	A100
GPUs per CPU socket	1 to 4
GPUs per node	1 to 8
Multi-node capable	Yes
Multi-node interconnect	ConnectX6 (EDR IB)



**nVIDIA**