HP-CAST, November 16, 2013

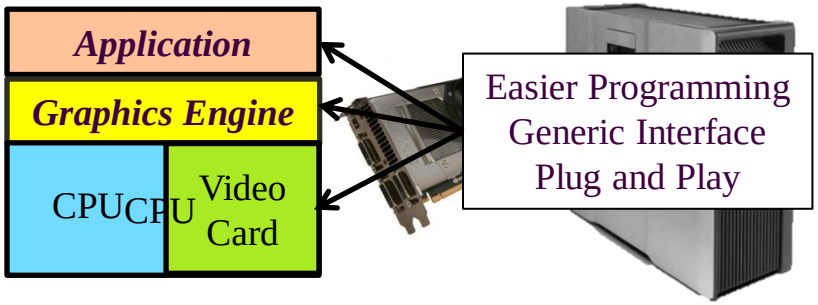
Dr. Ronald Young

Outstanding Performance
Using the latest NVIDIA GPUs
in HP Z820 Workstations and SL270 Servers

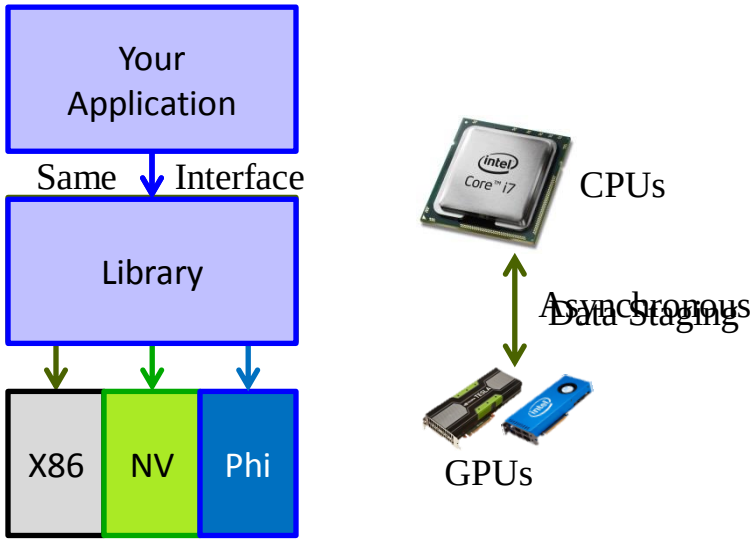
Focusing on:

1. **Plug-and-play libraries**
2. **FMSlib**
3. [Matrix]Warrior

What is Plug and Play?

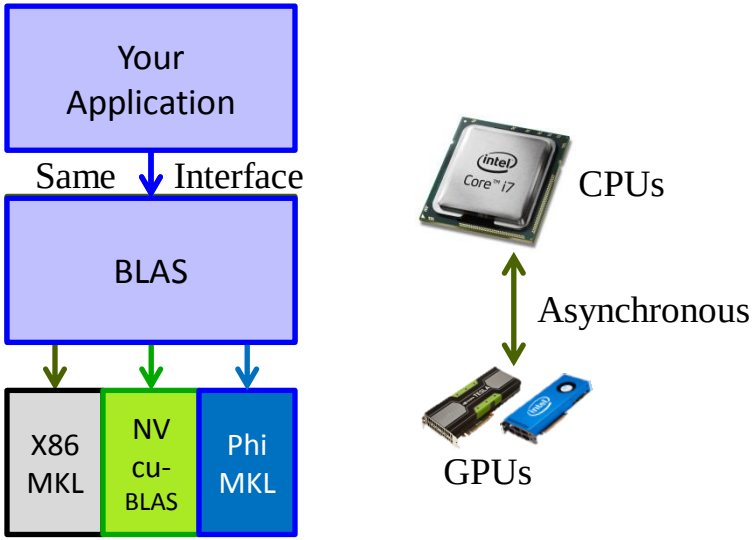


Part 1: Scientific Plug-and-Play Libraries



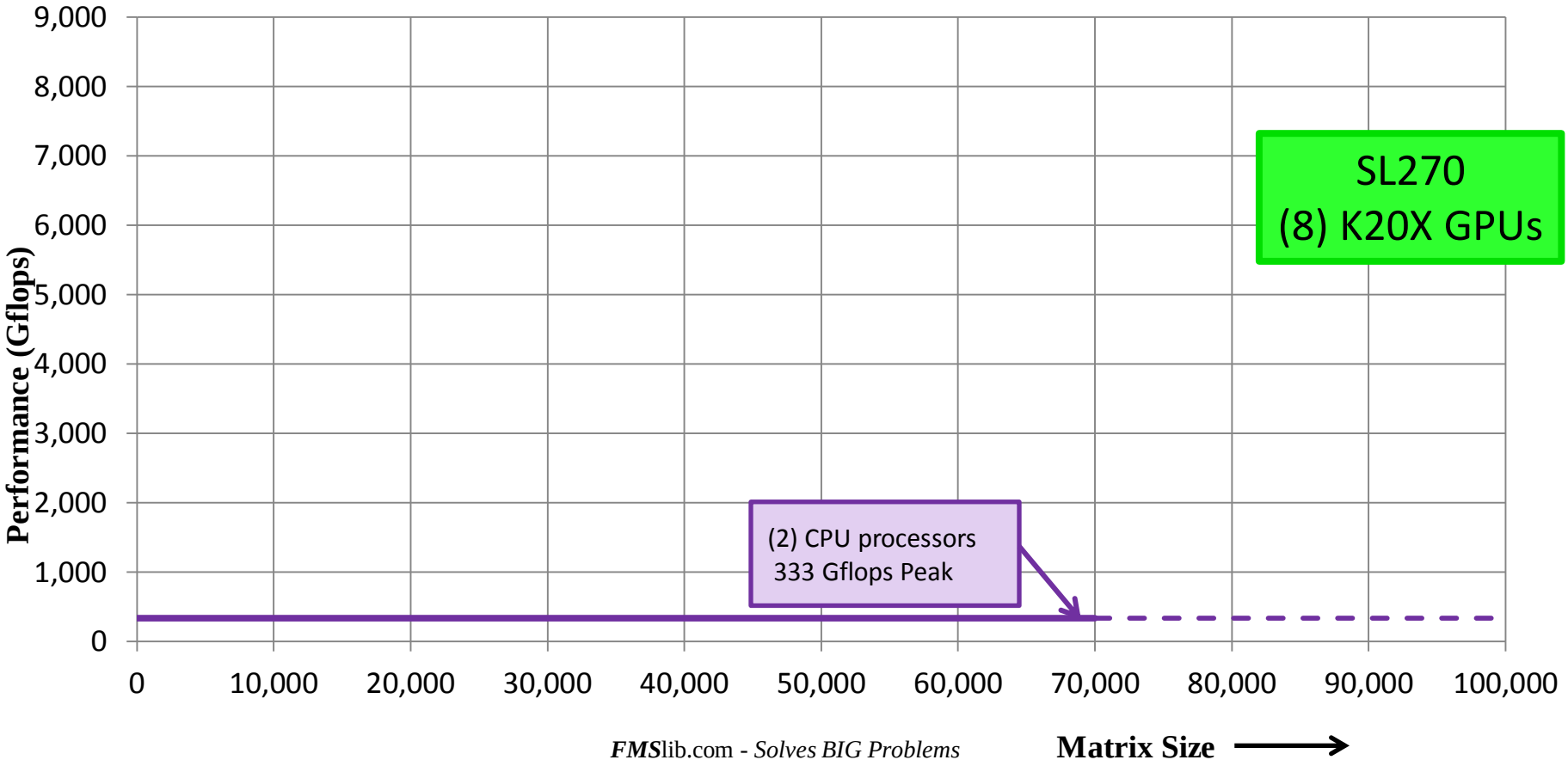
- Application runs on X86 (Linux | Windows) and calls library subroutines
- Want accelerator performance benefits (floating point, massively parallel)
- Want single version of your application
- No programming
- Accelerators have smaller faster memory (and maybe different instruction set)
- Plug-and-play libraries provide these benefits with no programming.
  - Automatically detects installed hardware
  - Divides work between CPUs and GPUs
  - Transfers data to/from GPUs asynchronously in parallel with processing.

Part 1: Plug-and-Play BLAS Library



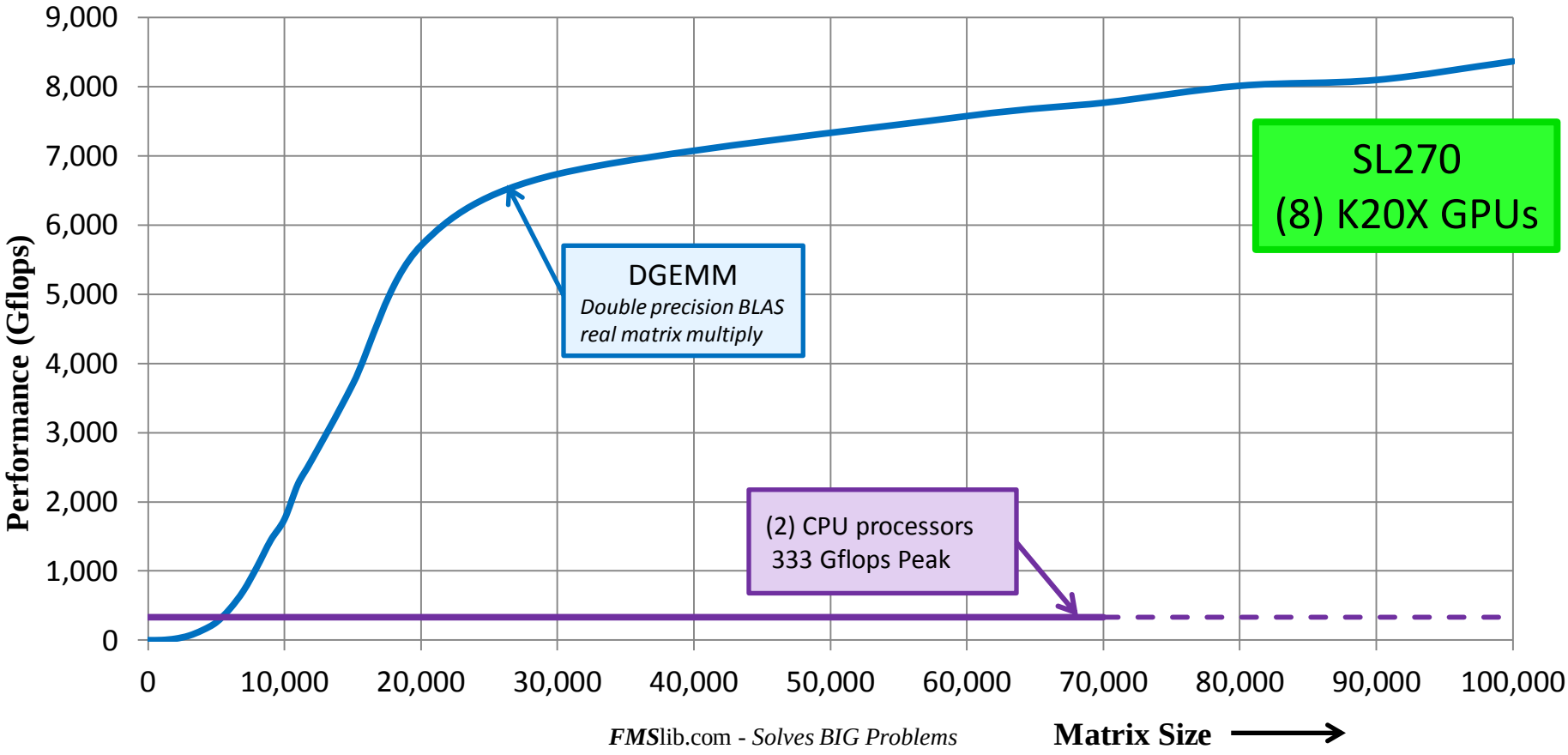
- Application runs on X86 (Linux | Windows) and calls BLAS subroutines
- Relink to PnP BLAS library
- Automatically detects hardware at startup and uses what is present.
- Automatically divides work between CPUs and GPUs, adjusted at runtime.
- Transfers data to/from GPUs in parallel with processing.
- Uses latest version of MKL and cuBLAS
- Single version of your application.
- **Requires NO programming**

Part 1: Plug-and-Play BLAS *Performance*



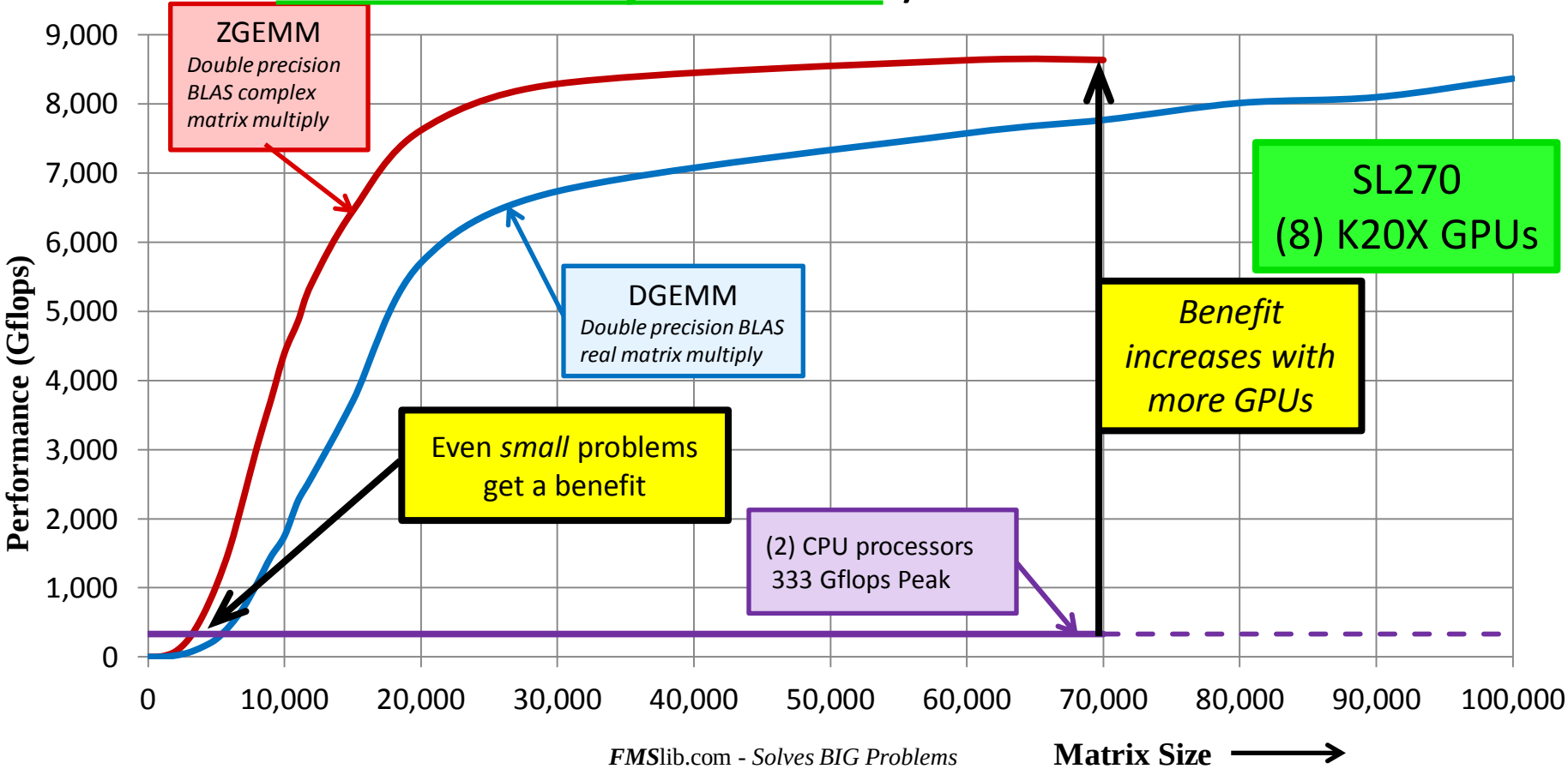
Part 1: Plug-and-Play BLAS *Performance*

Automatically increase your **BLAS** Performance

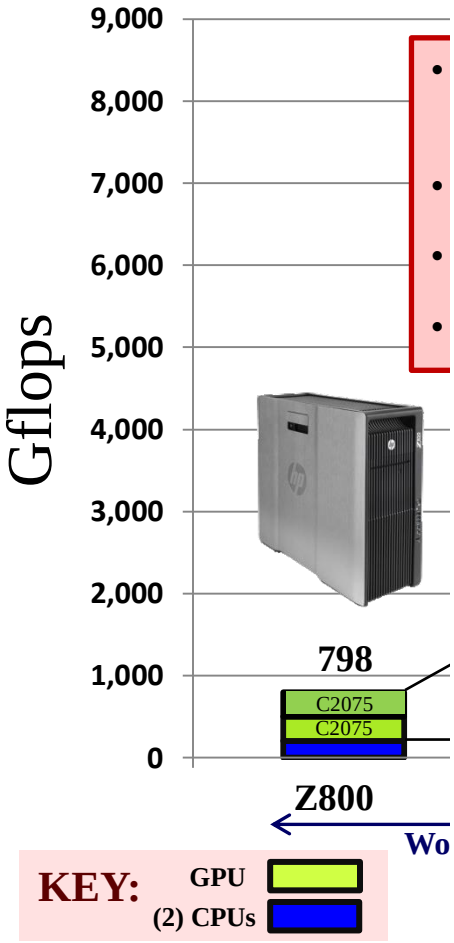


Part 1: Plug-and-Play BLAS *Performance*

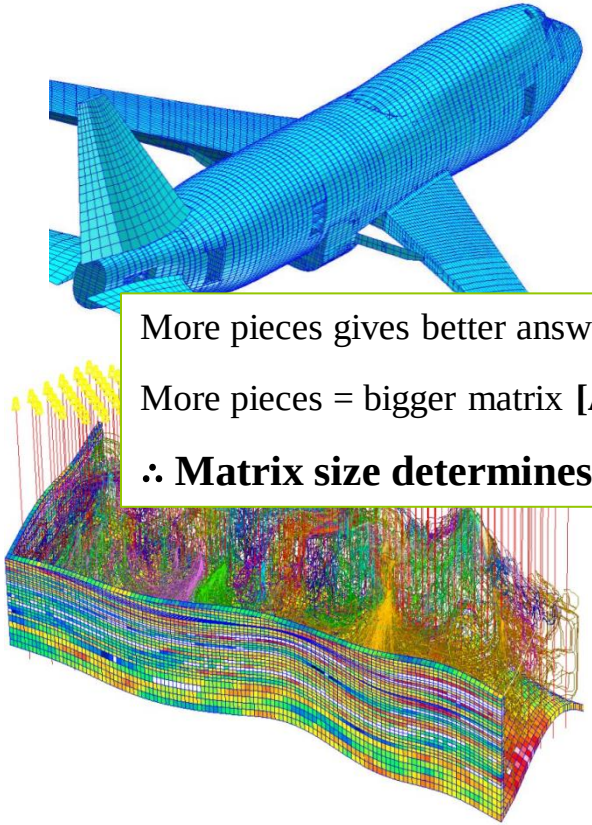
Automatically increase your **BLAS** Performance



Part 1: Plug-and-Play BLAS *Performance Trends*



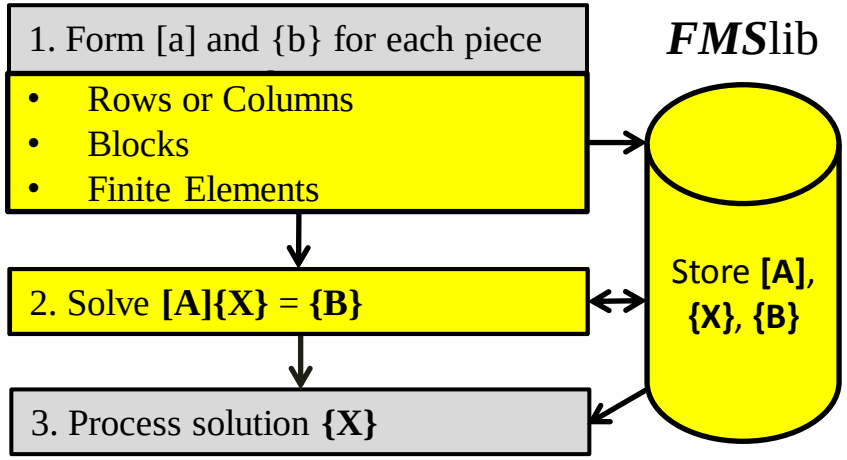
Part 2: FMSlib



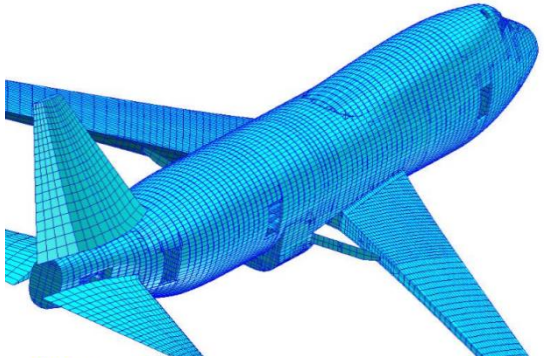
More pieces gives better answer

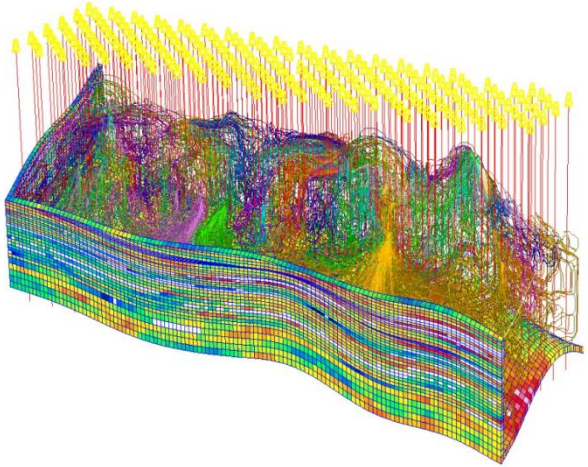
More pieces = bigger matrix **[A]**

∴ Matrix size determines accuracy



Part 2: FMSlib: Solving [A]{X}={B}





CPUs

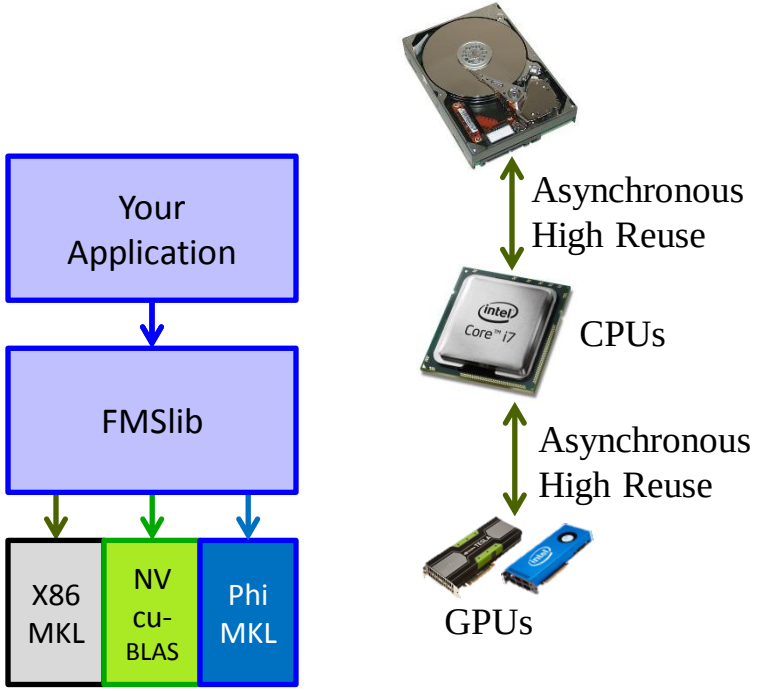
Data Reuse

GPUs

Approach	Equation	Accesses	Reuse	BLAS
Iterative	{E}=[A]{X}-{B}	Many	Low	L2(MV)
Direct	[A]{X}={B}	Once	High	L3(MM)

- Data movement uses power (Not + *)
- GPU perform best with high data reuse
- Port the science, not the code.

Part 2: FMSlib *Plug-and-play*

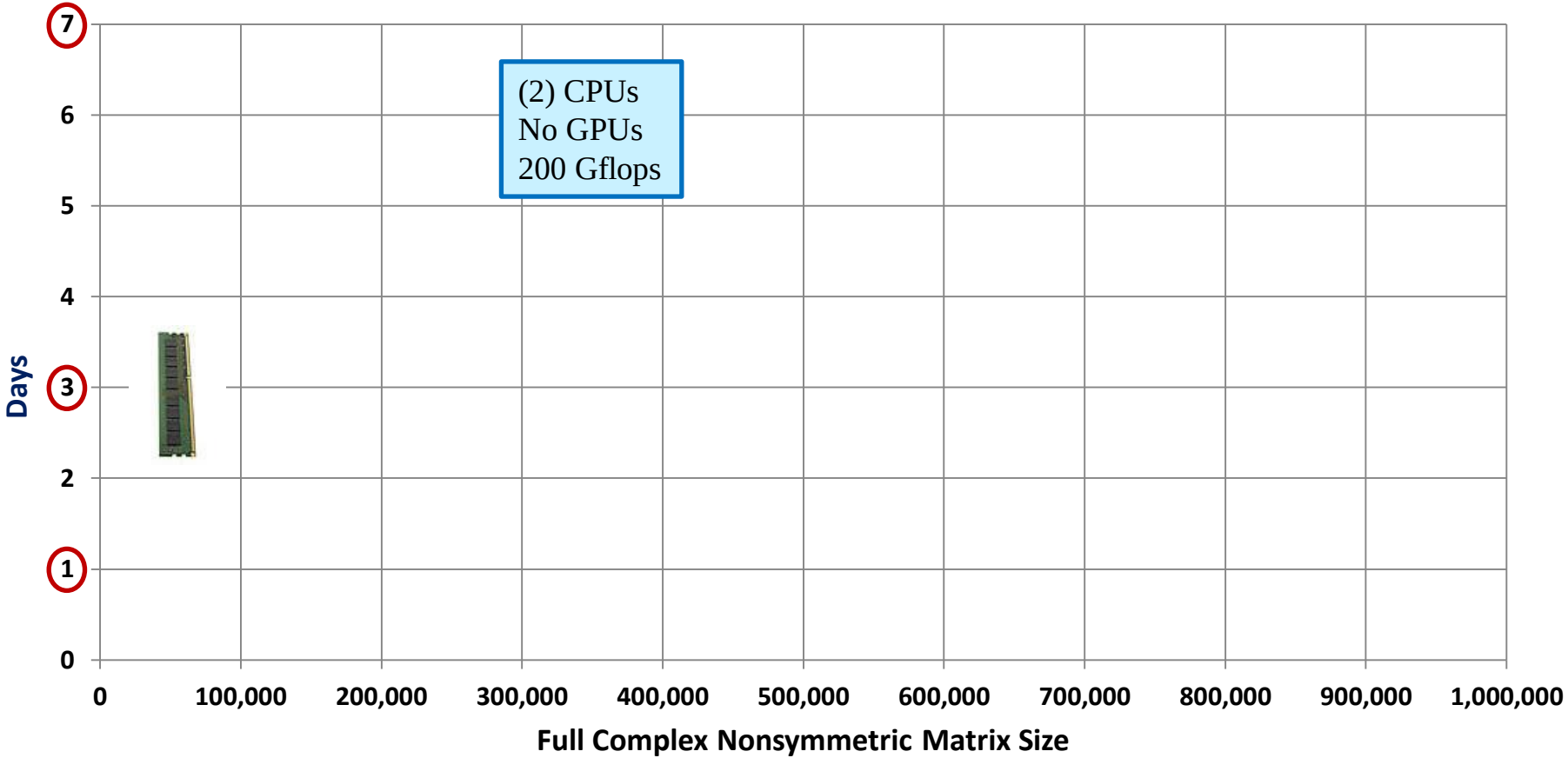


- Application runs on X86 (Linux | Windows) and calls FMSlib subroutines
- FMSlib used disk to solve extremely large problems.
- Transfers blocks of data to/from disk in parallel with processing

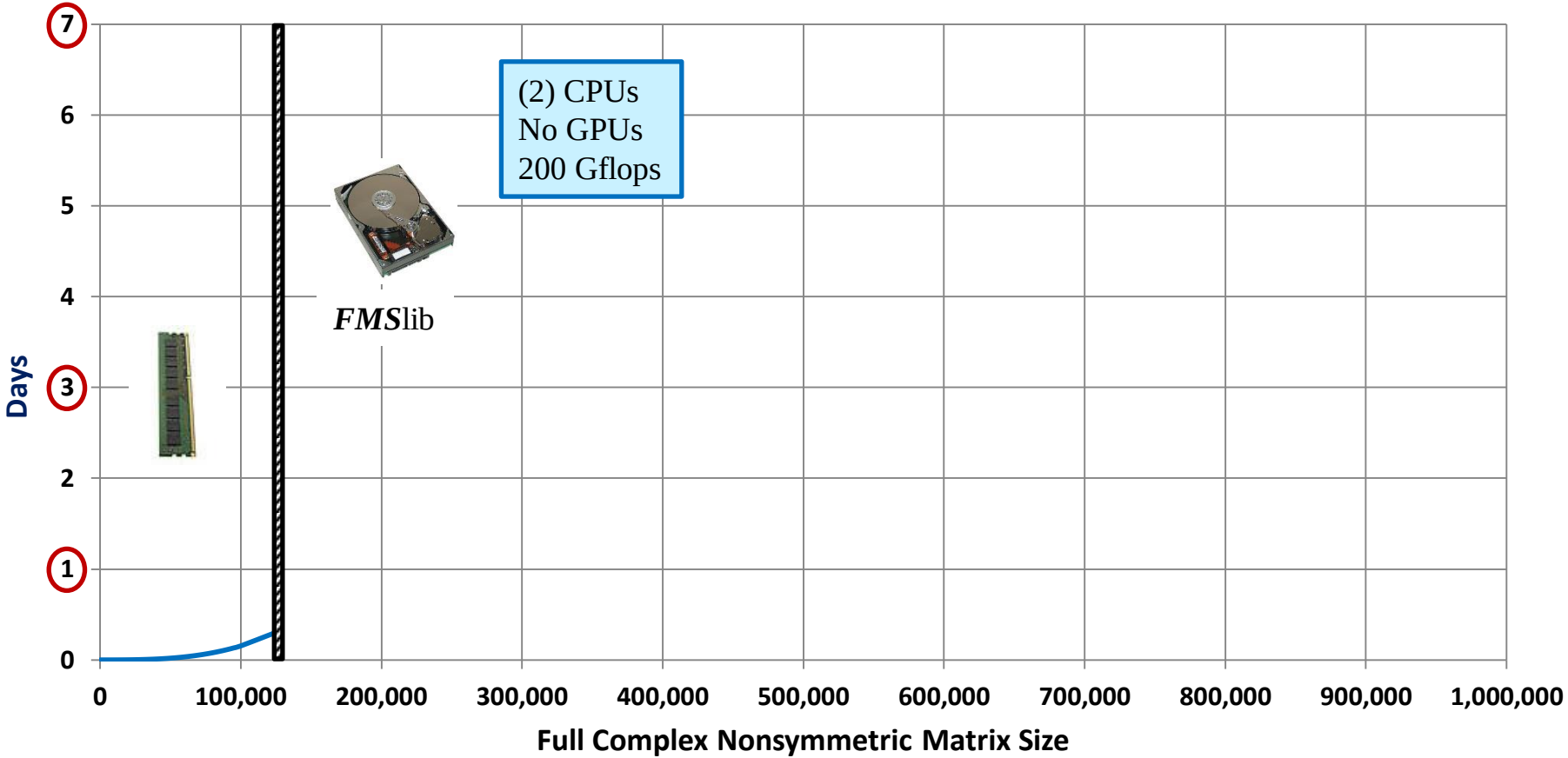
ADD GPUs

- Automatically detects hardware at startup and uses what is present.
- Automatically divides work between CPUs and GPUs, adjusted at runtime.
- Transfers data to/from GPUs in parallel with processing.
- Uses latest version of MKL and cuBLAS
- Single version of your application.
- **Requires NO additional programming**

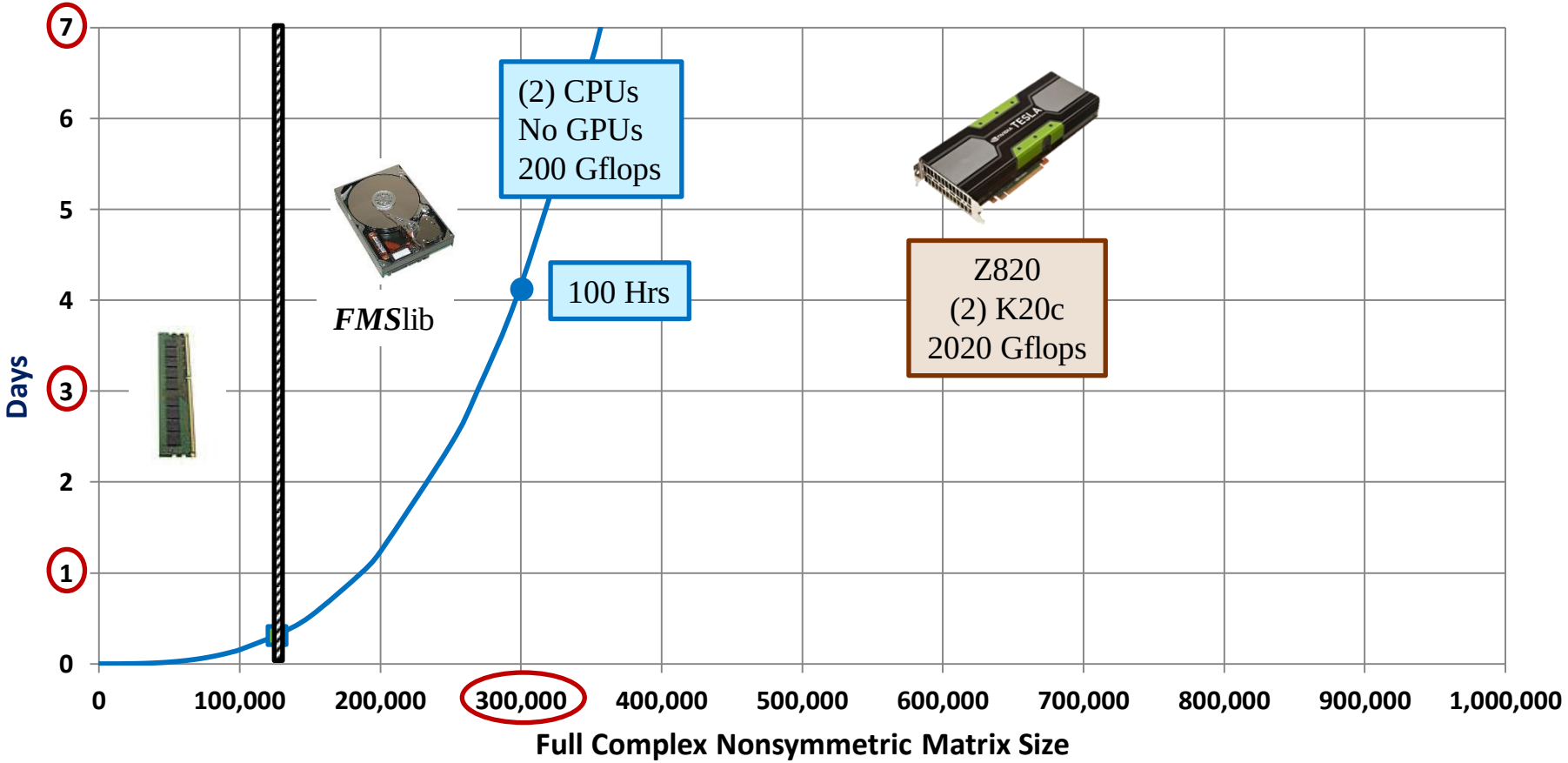
Part 2: FMSlib *Performance*



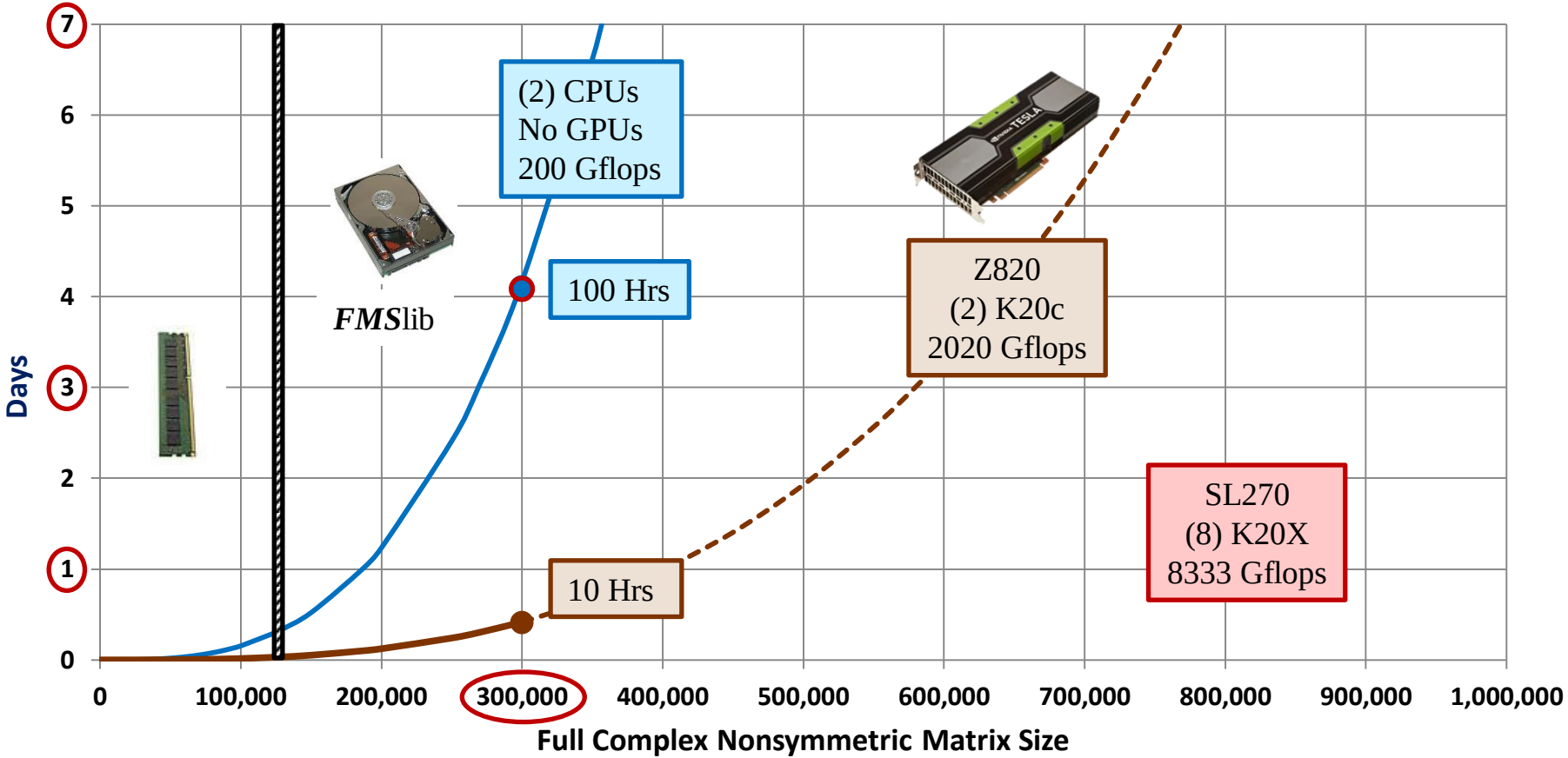
Part 2: FMSlib *Performance*



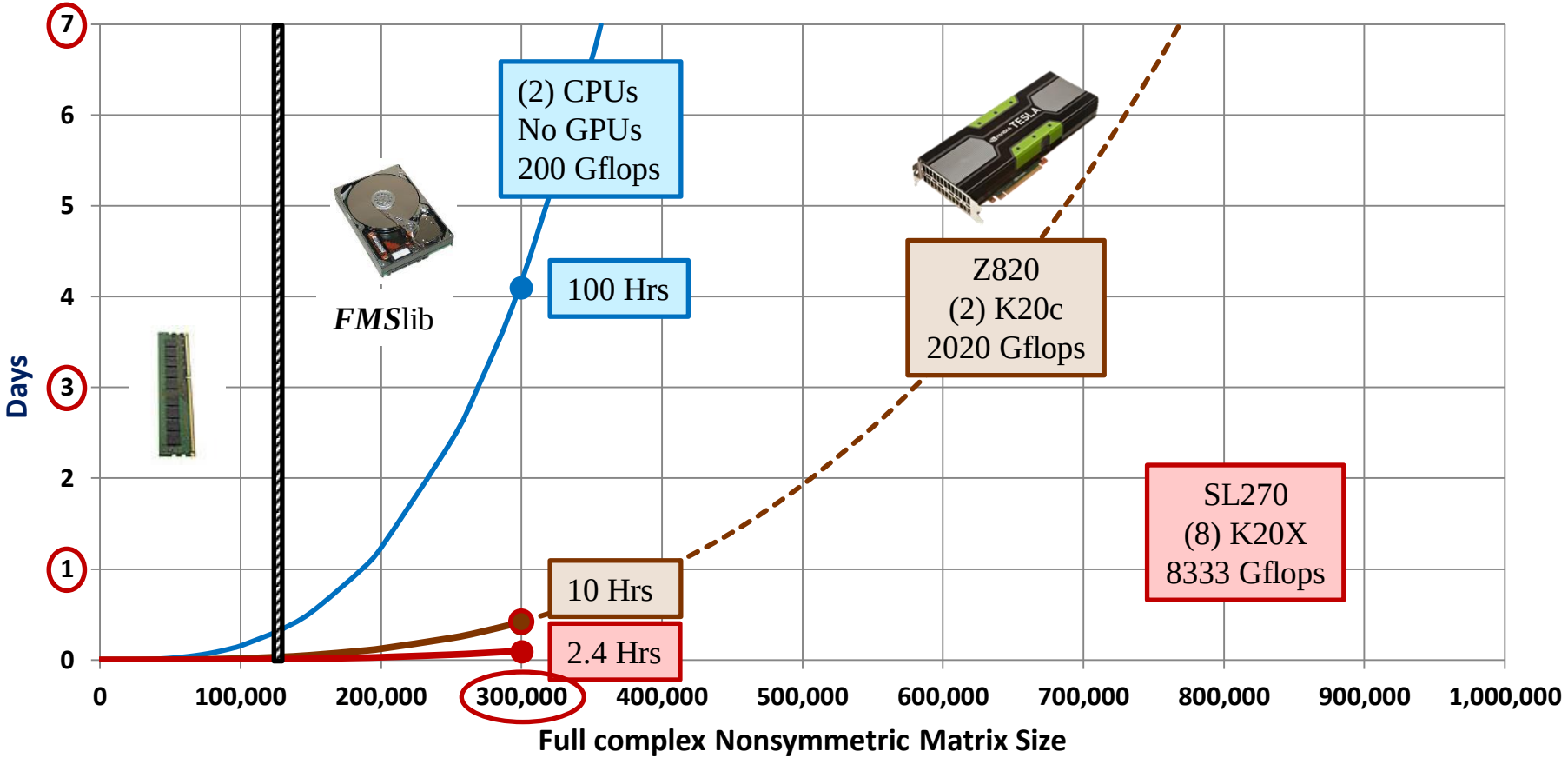
Part 2: FMSlib *Performance*



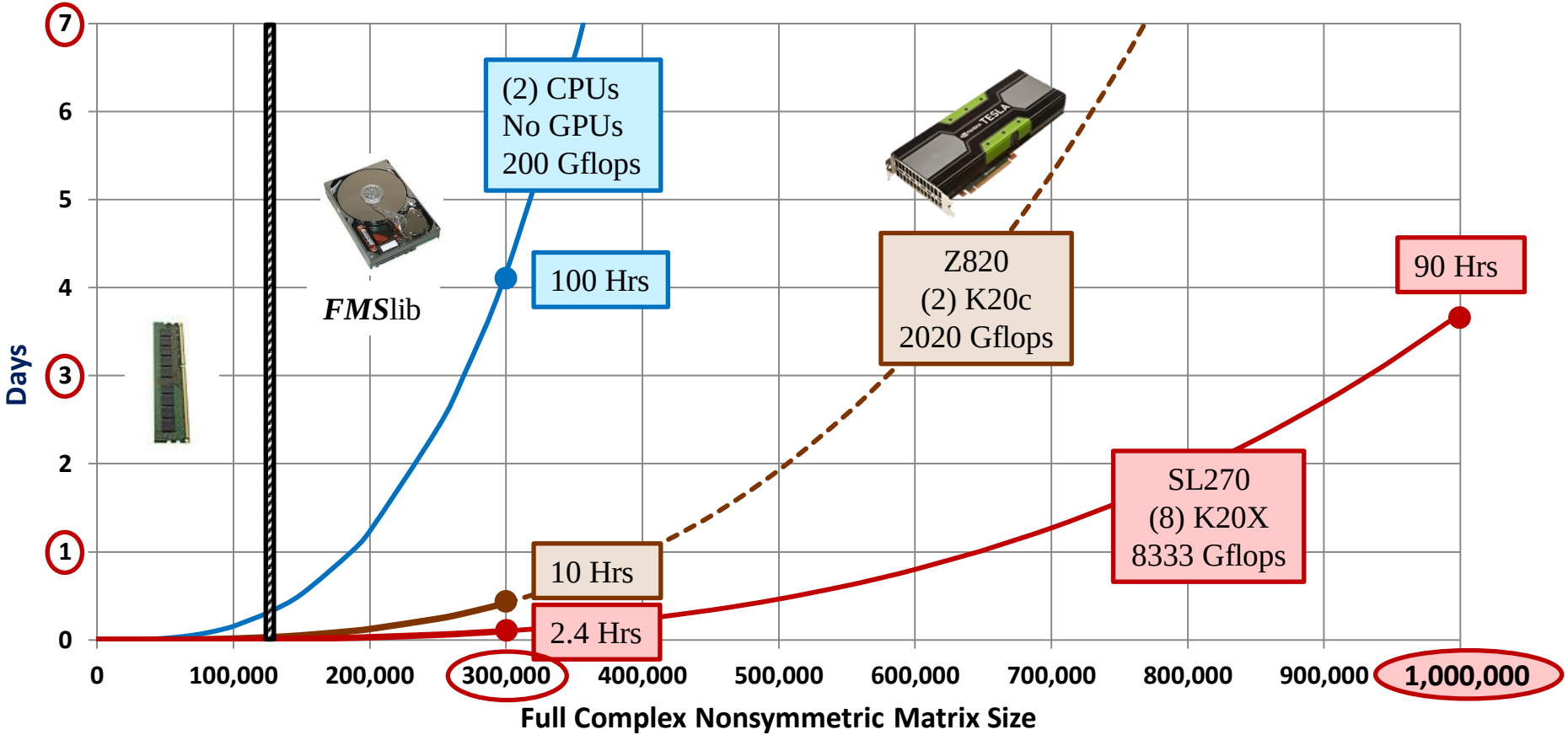
Part 2: FMSlib *Performance*



Part 2: FMSlib *Performance*



Part 2: FMSlib *Performance*



***FMS*lib Performance History 1978-present**

Machine	Year	Flops	$/Gflop
DEC VAX	**1978**	97,000	$2,000,000,000

FMSlib is Plug-and-play

SL270

(8) NVIDIA GPUs	**2013**	8,333,000,000,000	$12

86 Million times faster!
166 Million times cheaper!

Part 2: FMSlib *Performance Monitoring*

WEB pages generated dynamically during runtime

Part 2: FMSlib *Performance Monitoring - CPUs*

Main | **CPUs** | GPUs | Memory | Disks | Files | Software | Calls | Parameters | Performance | Usage | Links | Help

CPU Description

Number of processing cores	16
Cores used by FMS	16
Cores per processor	8
Minimum Clock Speed (Ghz)	2.594
Maximum Clock Speed (Ghz)	2.594
Processor Model	Genuine Intel(R) CPU @ 2.60GHz

Properties for GPU's 0 to 1 Used by FMS

GPU Number	RTL	0	1
	NVML	0	1
GPU Temperature		36 C	41 C
Fan Speed		30 %	30 %
Utilization	Gpu	0 %	0 %
	Memory	0 %	0 %
Memory Usage (Mbytes)	Total	5032	5032
	Used	13	13
	Free	5018	5018
Clocks (Cur./Max) (MHz)	Graphics	324/758	324/758
	SM	324/758	324/758
	Memory	324/2600	324/2600
Power (milliwatts)	State	P8	P8
	Draw	13401	12830
Clocks Throttle Reasons		GPU Idle	GPU Idle
ECC Enabled		Yes	Yes
ECC Errors Volatile	Single Bit	0	0
	Double Bit	0	0
ECC Errors Aggregate	Single Bit	0	0
	Double Bit	0	0
Compute Mode		NonExclusive	NonExclusive
Device Type (RTL)		Tesla K20c	Tesla K20c
Product Name (NVLM)		Tesla K20c	Tesla K20c
Serial Number		0333712000480	0333712000473

All information available from
- Run Time Library (RTL)
- Nvidia Management Library (NVML)

lves BIG Problems

CPUs Used		All	All
PCI Bus ID (hex)		6	41
PCI Device ID (hex)		0	0
PCI	Bus	6	41
	Device	0	0
	Domain	0	0
	Device Id	102210DE	102210DE
	Bus Id	0000:06:00.0	0000:41:00.0
	Sub System Info	982103C	982103C
Capability(manor.minor)		(3.5)	(3.5)
Multiprocessors		13	13
Warp size		32	32
Cores		832	832
Clock Rate (GHz)		0.71	0.71
Global Memory (bytes)		5032706048	5032706048
Display Mode		Enabled	Enabled
Persistence Mode		Enabled	Enabled
VBIOS Version		80.10.11.00.08	80.10.11.00.08
Inforom Version	OEM	1.1	1.1
	ECC	3.0	3.0
	Power	N/A	N/A
Supported Clock Throttle Reasons		N/A	N/A
Compute Mode		Default	Default
Power Management		Enabled	Enabled
Power Limits (milliwatts)	Default	235000	235000
	Current	235000	235000
	Minimum	150000	150000
	Maximum	N/A	N/A

Page updated 1 times: Last Updated 15:13:02 Tue Feb 19 2013

Part 2: FMSlib *Performance Monitoring - Disks*

Main | CPUs | GPUs | Memory | **Disks** | Files | Software | Calls | Parameters | Performance | Usage | Links | Help

FMS Directory Link	Free GBytes Total GBytes	% Available	Access Type	Direction	Number Accesses	Avg. Transfer (MBytes)	Performance (MB/Sec.)
FMSDISK /data/FMSDISK	734.430 2200.253	33	Async.	Read	1353	7135	551
				Write	212	7073	490

Disk Performance by File Type

File Type			Number Accesses	Avg. Transfer (MBytes)	Performance (MB/Sec.)
[AL]	Async.	Read	687	7140	550
		Write	105	7140	491
[D]	Sync.	Write	1	0	865
[AU]	Async.	Read	665	7140	552
		Write	105	7140	489
SEG.	Sync.	Read	1	5	2232
		Write	1	5	786

Overall Disk Performance

Transfer Type		Total		Overall MBytes/Sec
		MBytes	Time(Sec.)	
Async.	Read	9653618	17525	551
	Write	1499452	3058	490
Sync.	Read	5	0	2232
	Write	11	0	816

Page updated 2 times: Last Updated 03:08:00 Wed Feb 20 2013

Transfer Rate (MB/Sec)

Main | CPUs | GPUs | Memory | Disks | Files | Software | Calls | Parameters | **Performance** | Usage | Links | Help

Performance Analysis for Subroutine CNDF , Time used=15.992

1% Complete

33% Other | 67% DIA

Performance (Gflops)

Routine	All	CPU's	GPU 1	GPU 2
Routine Overall	487	5	266	257

Legend

Undefined | Assembled | Factored | Reduced | Solved | Multiplied | Accessing | Computing

L1,1	A1,2	A1,3	A1,4
A2,1	A2,2	A2,3	A2,4
A3,1	A3,2	A3,3	A3,4
A4,1	A4,2	A4,3	A4,4

Times and Problem

Job Started	0	12:42:02 Wed Mar 13 2013	Equations	50000
Routine Started	43	12:42:46 Wed Mar 13 2013	Vectors	0
Current Time	59	12:43:02 Wed Mar 13 2013	Block Size	12544 X 12544
Estimated Completion	1056	12:59:38 Wed Mar 13 2013	Data Type	Complex

I/O Compute Ratio

Compute Rate (Gflops)	487
Read Rate (MBytes/Sec.)	0
Reuse	25088
IO Compute Ratio	0.00

Page updated 5 times: Automatically refreshed every 10 sec.: Last Updated 12:43:02 Wed Mar 13 2013

Subroutine

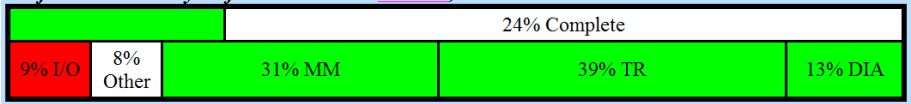
Status Bar

% Time

Performance

Application specific graphic

Performance Analysis for Subroutine CNDF ; Time used=02:41



Performance (Gflops)

Routine	All	CPU's	GPU 1	GPU 2
Routine Overall	605	24	308	304

Legend

Undefined	Assembled	Factored	Reduced	Solved	Multiplied	Accessing	Computing

L1,1	U1,2	U1,3	A1,4
L2,1	L2,2	M2,3	A2,4
L3,1	L3,2	A3,3	A3,4
A4,1	A4,2	A4,3	A4,4

Times and Problem

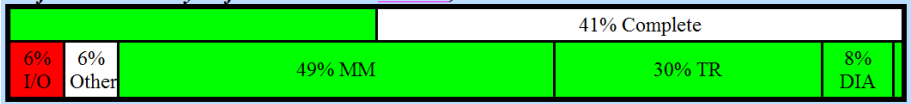
Job Started	0	12:42:02 Wed Mar 13 2013	Equations	50000
Routine Started	43	12:42:46 Wed Mar 13 2013	Vectors	0
Current Time	204	12:45:26 Wed Mar 13 2013	Block Size	12544 X 12544
Estimated Completion	701	12:53:43 Wed Mar 13 2013	Data Type	Complex

I/O Compute Ratio

Compute Rate (Gflops)	605
Read Rate (MBytes/Sec.)	0
Reuse	25088
IO Compute Ratio	0.00

Page updated 22 times: Automatically refreshed every 12 sec.: Last Updated 12:45:26 Wed Mar 13 2013

Performance Analysis for Subroutine CNDF ; Time used=04:10



Performance (Gflops)

Routine	All	CPU's	GPU 1	GPU 2
Routine Overall	618	24	310	306

Legend

Undefined	Assembled	Factored	Reduced	Solved	Multiplied	Accessing	Computing

L1,1	U1,2	U1,3	A1,4
L2,1	L2,2	U2,3	A2,4
L3,1	L3,2	D3,3	A3,4
A4,1	A4,2	A4,3	A4,4

Times and Problem

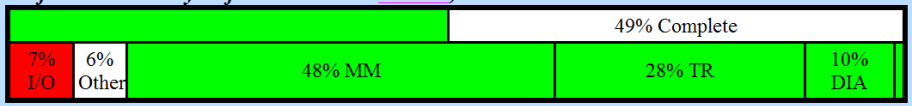
Job Started	0	12:42:02 Wed Mar 13 2013	Equations	50000
Routine Started	43	12:42:46 Wed Mar 13 2013	Vectors	0
Current Time	293	12:46:56 Wed Mar 13 2013	Block Size	12544 X 12544
Estimated Completion	652	12:52:55 Wed Mar 13 2013	Data Type	Complex

I/O Compute Ratio

Compute Rate (Gflops)	618
Read Rate (MBytes/Sec.)	0
Reuse	25088
IO Compute Ratio	0.00

Page updated 30 times: Automatically refreshed every 24 sec.: Last Updated 12:46:56 Wed Mar 13 2013

Performance Analysis for Subroutine CNDF ; Time used=05:05



Performance (Gflops)

Routine	All	CPU's	GPU 1	GPU 2
Routine Overall	618	24	311	307

Legend

Undefined	Assembled	Factored	Reduced	Solved	Multiplied	Accessing	Computing

L1,1	U1,2	U1,3	A1,4
L2,1	L2,2	U2,3	A2,4
L3,1	L3,2	L3,3	A3,4
L4,1	T4,2	A4,3	A4,4

Times and Problem

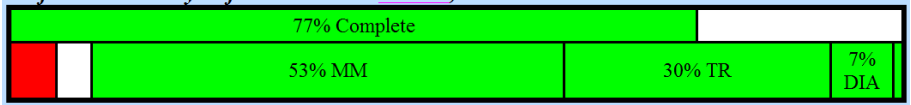
Job Started	0	12:42:02 Wed Mar 13 2013	Equations	50000
Routine Started	43	12:42:46 Wed Mar 13 2013	Vectors	0
Current Time	348	12:47:50 Wed Mar 13 2013	Block Size	12544 X 12544
Estimated Completion	657	12:53:00 Wed Mar 13 2013	Data Type	Complex

I/O Compute Ratio

Compute Rate (Gflops)	618
Read Rate (MBytes/Sec.)	0
Reuse	25088
IO Compute Ratio	0.00

Page updated 36 times: Automatically refreshed every 23 sec.: Last Updated 12:47:50 Wed Mar 13 2013

Performance Analysis for Subroutine CNDF ; Time used=07:27



Performance (Gflops)

Routine	All	CPU's	GPU 1	GPU 2
Routine Overall	633	24	315	312

Legend

Undefined | Assembled | Factored | Reduced | Solved | Multiplied | Accessing | Computing

L1,1	U1,2	U1,3	U1,4
L2,1	L2,2	U2,3	U2,4
L3,1	L3,2	L3,3	M3,4
L4,1	L4,2	L4,3	A4,4

Times and Problem

Job Started	0	12:42:02 Wed Mar 13 2013	Equations	50000
Routine Started	43	12:42:46 Wed Mar 13 2013	Vectors	0
Current Time	491	12:50:13 Wed Mar 13 2013	Block Size	12544 X 12544
Estimated Completion	619	12:52:22 Wed Mar 13 2013	Data Type	Complex

I/O Compute Ratio

Compute Rate (Gflops)	633
Read Rate (MBytes/Sec.)	0
Reuse	25088
IO Compute Ratio	0.00

Page updated 52 times: Automatically refreshed every 23 sec.: Last Updated 12:50:13 Wed Mar 13 2013

Performance Analysis for Subroutine CNDF ; Time used=09:44

COMPLETED				
6% I/O		57% MM	25% TR	7% DIA

Performance (Gflops)

Routine	All	CPU's	GPU 1	GPU 2
Routine Overall	633	24	315	311

Legend

Undefined	Assembled	Factored	Reduced	Solved	Multiplied	Accessing	Computing

L1,1	U1,2	U1,3	U1,4
L2,1	L2,2	U2,3	U2,4
L3,1	L3,2	L3,3	U3,4
L4,1	L4,2	L4,3	L4,4

Times and Problem

Job Started	0	12:42:02 Wed Mar 13 2013	Equations	50000
Routine Started	43	12:42:46 Wed Mar 13 2013	Vectors	0
Current Time	627	12:52:30 Wed Mar 13 2013	Block Size	12544 X 12544
Estimated Completion	627	12:52:30 Wed Mar 13 2013	Data Type	Complex

I/O Compute Ratio

Compute Rate (Gflops)	633
Read Rate (MBytes/Sec.)	341
Reuse	25088
IO Compute Ratio	1.69

Page updated 64 times: Automatically refreshed every 10 sec.: Last Updated 12:52:30 Wed Mar 13 2013

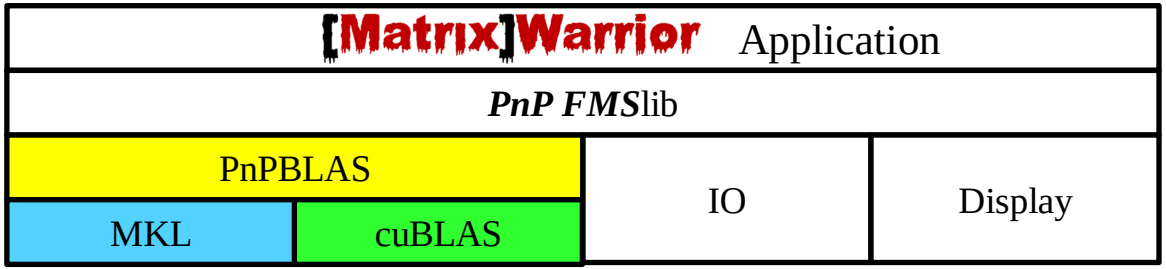
Part 2: FMSlib ***BENCHMARKS***

GPUs	Total	CPUs	GPU 1	GPU 2	GPU 3	GPU 4	GPU 5	GPU 6	GPU 7	GPU 8
C2075	798	208	305	305						
K20c	2,020	216	921	921						
M2070	2,331	71	297	297	297	295	295	295	300	299
M2090	3,315	154	401	413	413	410	401	403	404	403
K20	3,859	157	945	943	947	945				
K20X	8,333	Not Used	1068	1068	1068	1067	1069	1068	1068	1068

(2) K20c

(8) K20X

Part 3: [MatrixWarrior] - *An application which benchmarks and demonstrates FMSlib*



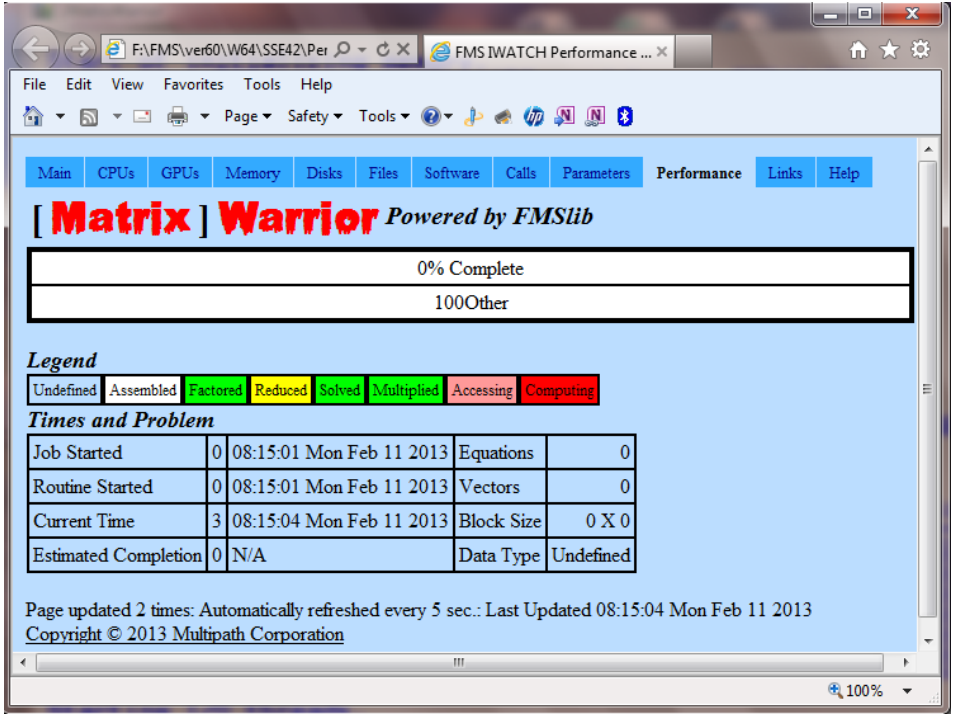
1. Form **[A]**, **{B}** with test data
2. Uses FMSlib to solve **[A]{X}={B}**

[Matrix]Warrior is an application that can:

1. Help setup and configure a machine
2. Demonstrate machine performance
   (Easily change CPUs, GPUs, Memory, Disks)
3. Benchmark different machine configurations
4. Burn-in a new machine
5. Serve as a hardware acceptance test
6. Validate new hardware designs
7. Demonstrate state-of-the-art performance for direct matrix solutions.

Running [Matrix]Warrior

Do you want to open your browser to view them(y or n)[Y]?>



Running [Matrix]Warrior

Do you want to open your browser to view them(y or n)[Y]?>

Enter the type of matrix you want to solve (1 to 5)
1 = Real Symmetric
2 = Real Nonsymmetric
3 = Complex Hermitian
4 = Complex Symmetric
5 = Complex Nonsymmetric
(1-5)[5]>

	Storage	Processing
1. Real Symmetric	1	1
2. Real Nonsymmetric	2	2
3. Complex Hermitian	2	4
4. Complex Symmetric	2	4
5. Complex Nonsymmetric	4	8

Running [Matrix]Warrior

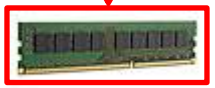
Do you want to open your browser to view them(y or n)[Y]?>

Enter the type of matrix you want to solve (1 to 5)

1 = Real Symmetric
2 = Real Nonsymmetric
3 = Complex Hermitian
4 = Complex Symmetric
5 = Complex Nonsymmetric

(1-5)[5]>

Top 500 Benchmark



Do you want to use disks(y or n)[N]?>

Running [Matrix]Warrior

Do you want to open your browser to view them(y or n)[Y]?>

Enter the type of matrix you want to solve (1 to 5)
1 = Real Symmetric
2 = Real Nonsymmetric
3 = Complex Hermitian
4 = Complex Symmetric
5 = Complex Nonsymmetric
(1-5)[5]>

Computed from:
- Data Type & Matrix Symmetry (1-5)
- Storage Location (Memory, Disk)

Do you want to use disks(y or n)[N]?>

Enter the size of the matrix[12676]>5000

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ A_{21} & A_{22} & A_{23} & \dots & A_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ A_{n1} & A_{n2} & A_{n3} & \dots & A_{nn} \end{bmatrix}$$

Running [Matrix]Warrior

Do you want to open your browser to view them(y or n) **[Y]**?>

Enter the type of matrix you want to solve (1 to 5)
1 = Real Symmetric
2 = Real Nonsymmetric
3 = Complex Hermitian
4 = Complex Symmetric
5 = Complex Nonsymmetric
(1-5)**[5]**>

Do you want to use disks(y or n) **[N]**?>

Enter the size of the matrix **[12676]**>5000

Enter the number of solution vectors **[0]**>

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ A_{21} & A_{22} & A_{23} & \dots & A_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ A_{n1} & A_{n2} & A_{n3} & \dots & A_{nn} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ \dots \\ B_n \end{bmatrix}$$

Running [Matrix]Warrior

Do you want to open your browser to view them(y or n) **[Y]**?>

Enter the type of matrix you want to solve (1 to 5)
1 = Real Symmetric
2 = Real Nonsymmetric
3 = Complex Hermitian
4 = Complex Symmetric
5 = Complex Nonsymmetric
(1-5)**[5]**>

Do you want to use disks(y or n) **[N]**?>

Enter the size of the matrix **[12676]**>5000

Enter the number of solution vectors **[0]**>

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} & \dots & A_{1n} \\ A_{21} & A_{22} & A_{23} & \dots & A_{2n} \\ \dots & \dots & \dots & \dots & \dots \\ A_{n1} & A_{n2} & A_{n3} & \dots & A_{nn} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ \dots \\ X_n \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ \dots \\ B_n \end{bmatrix}$$

You may now alter any FMS Parameter.
When you are finished, type the letters RETURN
FMSSET>

More Information

fmslib.com

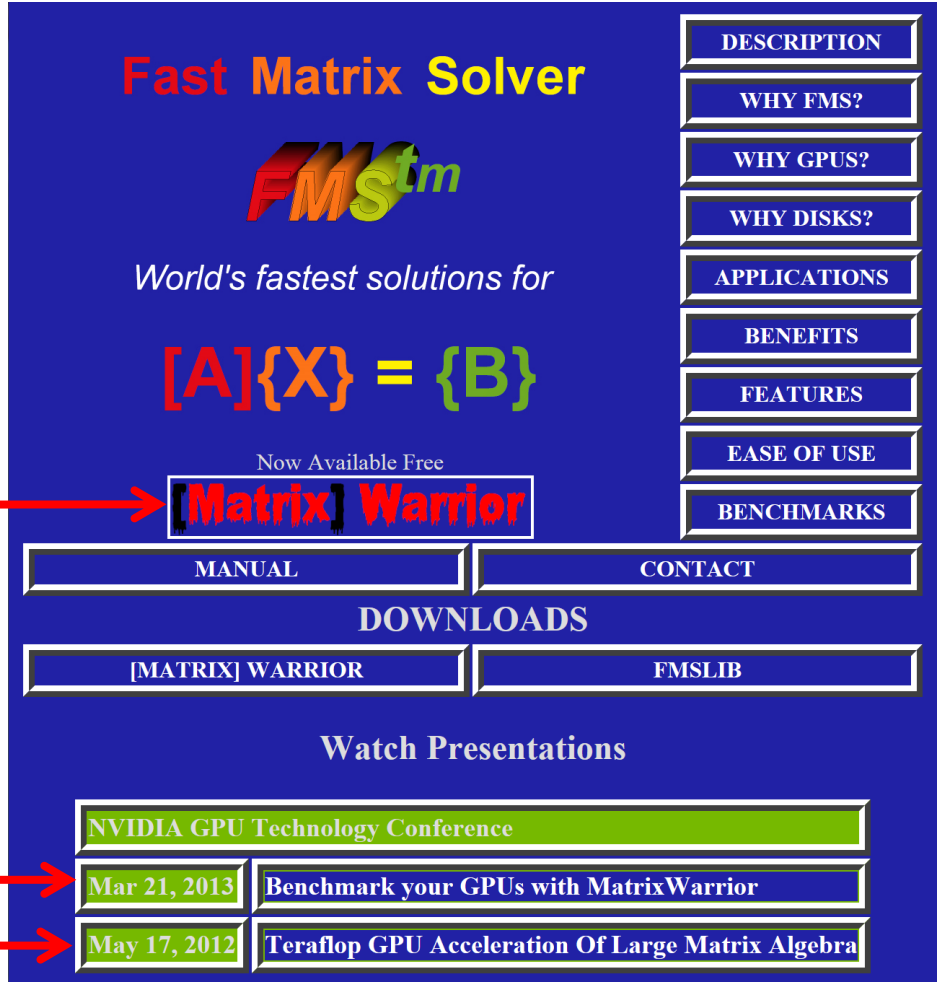
FREE Download

[Matrix]Warrior

NVIDIA GTC Presentations

[Matrix]Warrior

FMSlib



What's Available for FREE

- [Matrix]Warrior
- For FY2014
  - CPU Version of FMSlib
  - GPU version of FMSlib on configured Z820 and SL270