Implementation of Parallel Bitonic Sort using MPI and Intel TBB

Presented for CSE702 Fall 2021

Instructor: Dr. Russ Miller

Presenter: Zainul Abideen Sayed

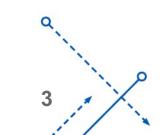
University at Buffalo The State University of New York



Why care about parallel sorting algorithms ?

Why parallel sorting?

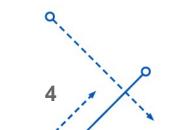
- <u>Sorting</u> is one of the most common operations in computation.
- The advancement in parallel hardware.
- Increasing nodes in a cluster and cores in a processor.
- Efficient utilization of resources.
- Therefore, good parallel sorting algorithms are needed.



Popular parallel sorting algorithms

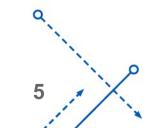
- Bitonic sort
- Sample sort
- Merge sort
- Quick sort
- Radix sort

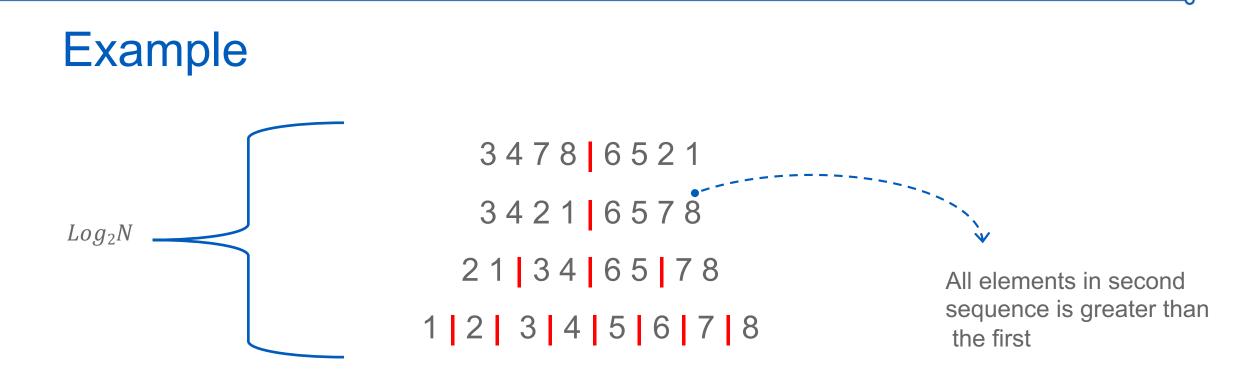
Bitonic sorting algorithm is based on bitonic sorting network. The key operation is based on the sorting network which converts a given sequence into a bitonic sequence and finally bitonic merge can produce a monotonically increasing or decreasing sequency.

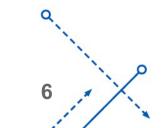


Bitonic Sort Principle

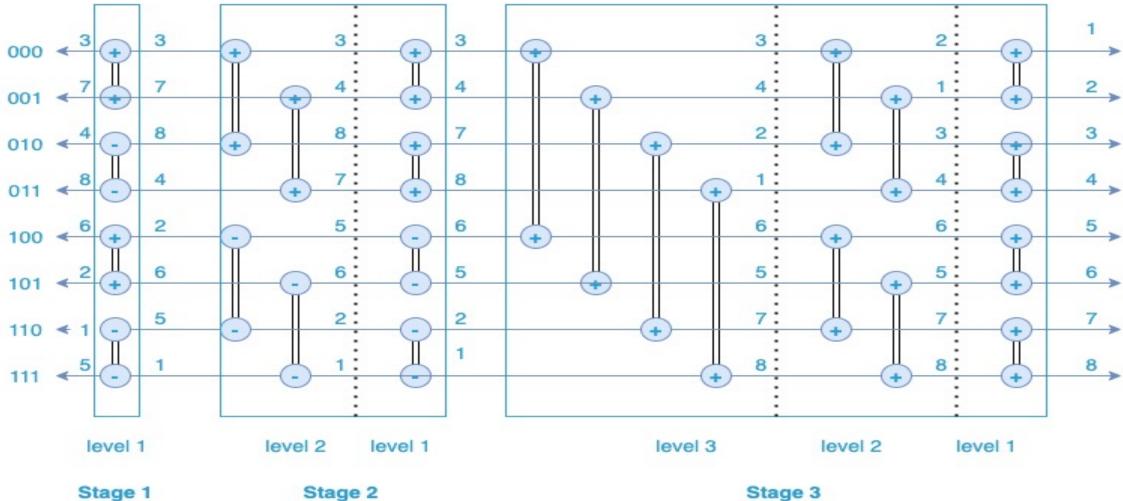
- Bitonic sequence <1, 2, 4, 7, 6, 0> <8, 9, 2, 1, 0, 4> <0, 4, 8, 9, 2, 1> <3, 4, 7, 8, 6, 5, 2, 1>
- Let s = <a0, a1, ..., an-1>
- $s_1 = \{ \min(a_0, a_{n/2}), \min(a_1, a_{n/2+1}), \dots, \min(a_{n/2-1}, a_{n-1}) \}$
- $s_2 = \{ \max(a_0, a_{n/2}), \max(a_1, a_{n/2+1}), \dots, \max(a_{n/2-1}, a_{n-1}) \}$
- In sequence s₁, there is an element b_i = min{ ai, a_{n/2+i} } such that all the elements before b_i are from the increasing part of the original sequence and all the elements after b_i are from the decreasing part.
- Opposite case for b_i = max{ ai , a_{n/2+i} }



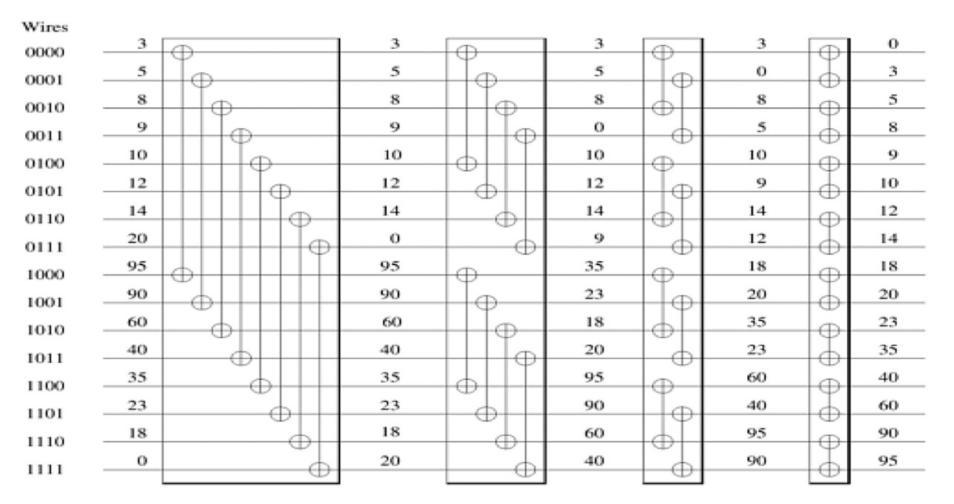








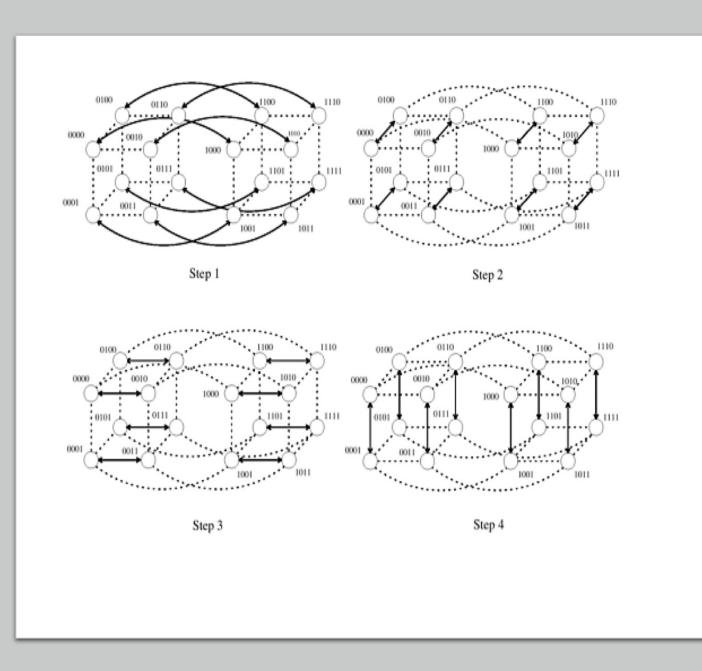
Example: 16 lines



Example: 16 lines

Hypercube

*Introduction to Parallel Computing 2nd Edition, Ananth Grama



Algorithm

COMPARE SECTION

```
procedure BITONIC_SORT(label, d)
begin
for i := 0 to d - 1 do
for j := i downto 0 do
    if (i + 1) st bit of label != j th bit of label then
        comp_exchange_max(j);
    else
        comp_exchange_min(j);
end BITONIC_SORT
```

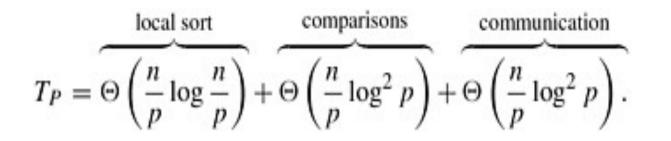
Complexity (1 + log n) (log n) / 2

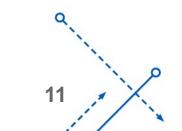
$T_p = \theta(n \log 2n))$ 0 Ω 0 comp_exchange_min(0); 0 comp_exchange_max(1); 2 1 2 comp_exchange_max(0); 0 comp exchange max(2); 3 2 comp exchange min(1); 3 1

3 0 comp_exchange_min(0);

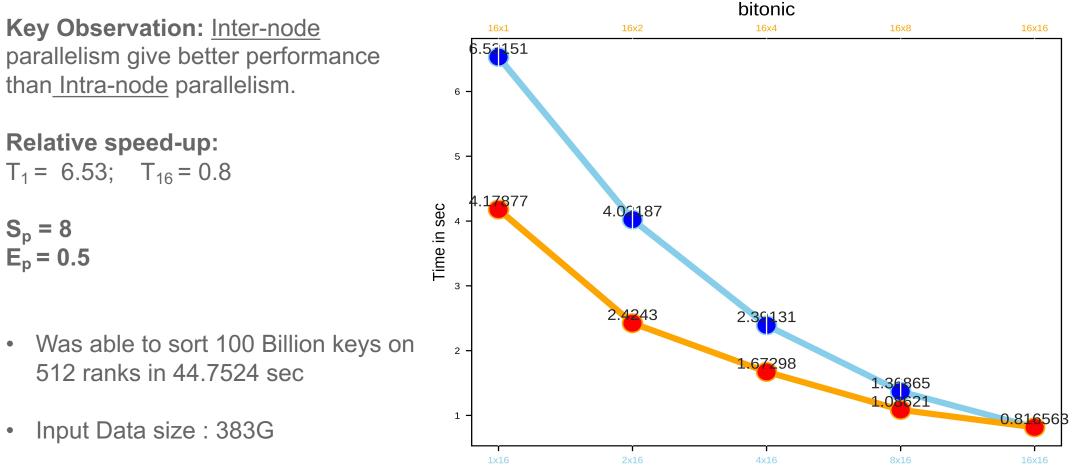
More on complexity...

- N/P Block of data per processor
- Fast sequential sort
 - Merge sort $\theta((n/p) \log(n/p))$
- Bitonic Merge
 - $\theta(\log 2 p)$





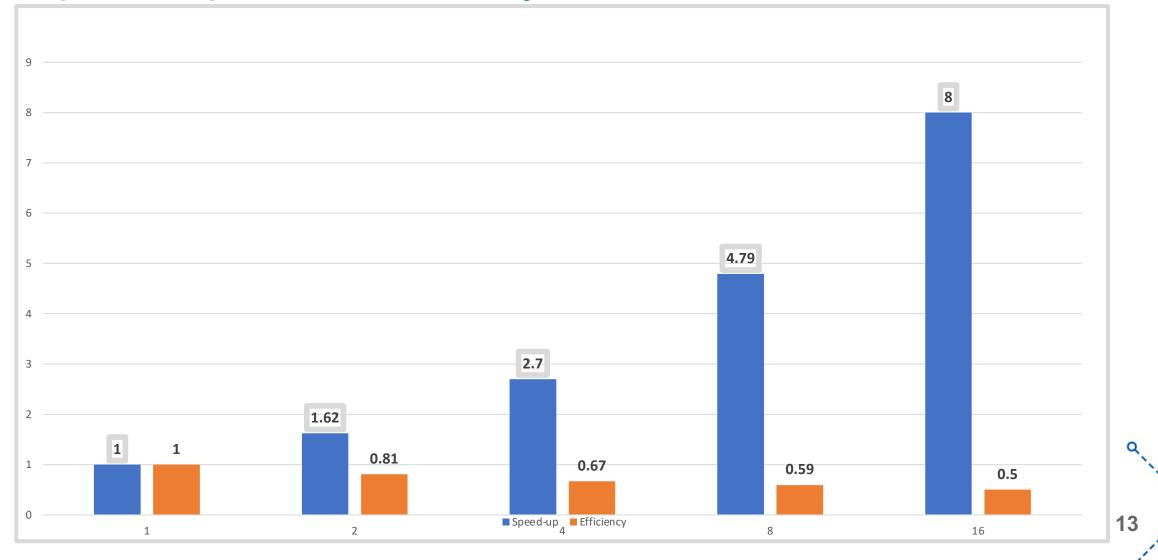
Results for 1 Billion keys

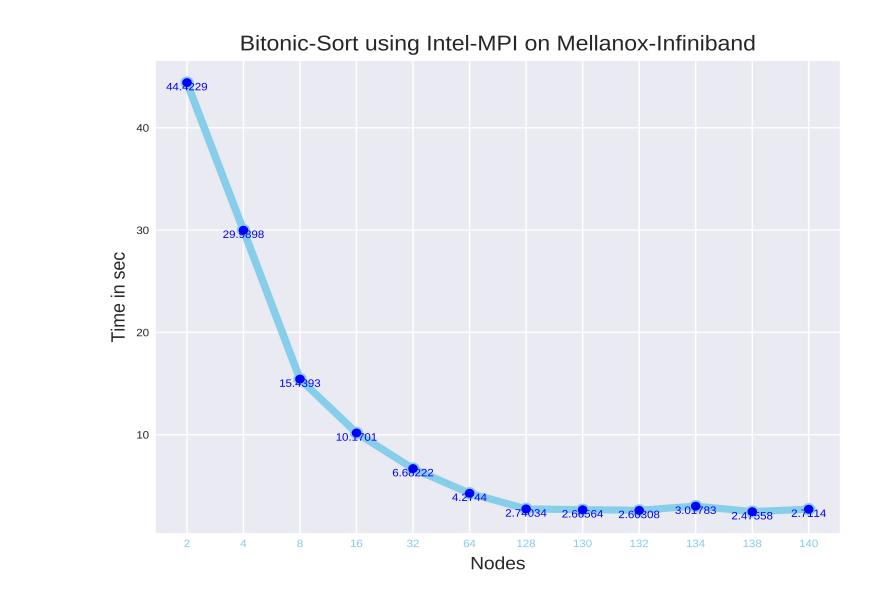


NodesXCores



Speed-up and Efficiency



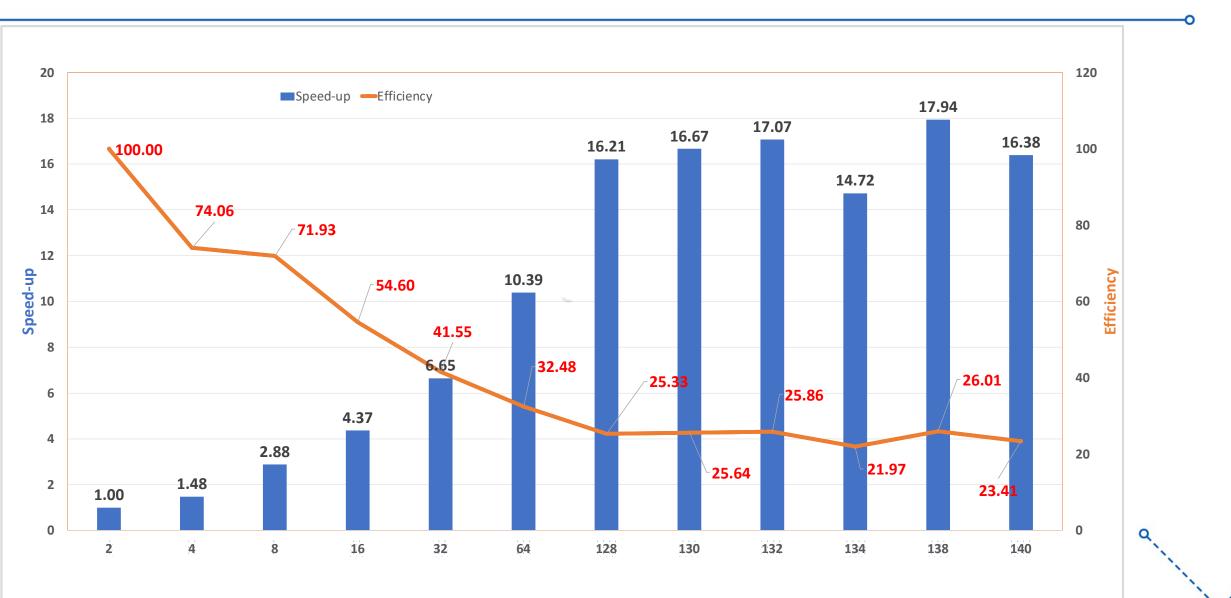


1 /

Results for 1 Billion keys

Observation:

- The code stops scaling after 128 Nodes.
- The execution time increases in some cases.
- However, It will be interesting to see the performance at 256 nodes.



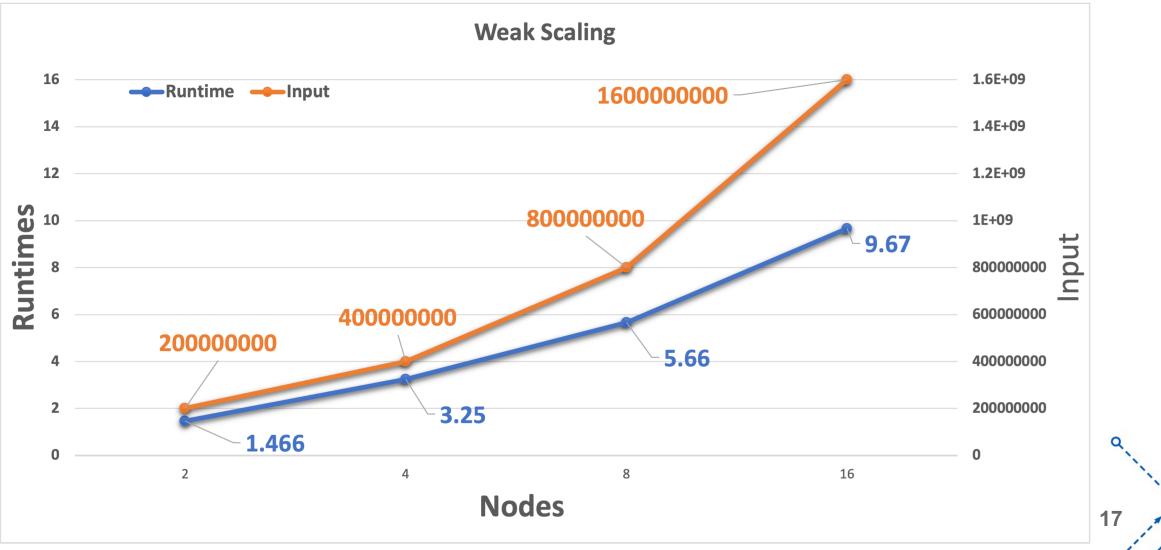
Amdahl's Law

The Law focuses on strong scaling where input remains constant, and we increase the processors expecting the runtime to reduce in proportion to number of processors added maintaining reasonable efficiency.

$$S_p = \frac{1}{\beta + \frac{(1-\beta)}{p}}$$
 where β is the serial part of the code which cannot be parallelized.

- In the above experiment: $(S_p = 16, p = 140)$ Therefore, β comes to = 5.5%.
- So according the Amdahl's Law 5.5% of code will never be parallelized.
- For Input size of 1 Billion keys we are able to strongly scale upto 32 nodes. After that the efficiency decrease dramatically.

Gustafson's law



Gustafson's law

As we increase the processor, we are able to solve bigger and bigger problems thus, achieving weak scaling

$$S_p = p - \alpha (p - 1)$$

$$\alpha = \frac{T_{seq}}{T_{seq} + Tpar}$$

Substituting the values from above experiment : (T_{seq} = 1.4, T_{par} = 9.6, α = 0.127)

$$S_p = 14.05$$

Therefore, the algorithm is weakly scalable at 16 nodes as N:P are in ratio.

Running without TBB (Inter MPI Tracer)

	•														· · · · · ·							
	0.5 s	1.0 s	1.5 s	2.0 s	2.5 s	3.0 s	3.5 s	4.0 s	4.5 s	5.0 s		.5 s	6.0 s	6.5 s	5 7.0		.5 s	8.0		8.5 s	9.0 s	9.5 s
P0 Appl	ication	1	Application	1	1		1	1	MPI_I	Applic <mark>MF</mark> App	_		-	Applic			MPI_ApM		_	M <mark>Applicati</mark>	_	. <mark>M</mark> App <mark>M</mark>
P1 Appl	ication	MPL Ba	Application									V IP <mark>I Applicatio</mark>		Applic	Application	1. Application		Applic	Applicati		MPApplicati	
P2 Appl	ication		Application						MP	Applic <mark>M</mark> Ap	X				Δ					I	ΥY	
										X		Δ	VI	1	1		X	Į.		W	Δ	X
	ication		Application							Applic <mark>M</mark> App		_		l W						WV	-	
'4 Appl	ication	MPI_Ba	Application						MPI	Applic: <mark>M</mark> Ap	<mark>/F</mark> Applic <mark>M</mark>	Applicati	Api <mark>MPI</mark>	Applic	Application	M <mark>Applicati</mark> M	Ap <mark>MPI</mark>	Applic	Applicati	Applicati	Applicatio	or <mark>MP</mark> Ap <mark>N</mark>
5 Appl	ication	MPI_B	Application						MP	Applic <mark>M</mark> Ap	4 <mark>6</mark> Applic <mark>N</mark>	Application	Ap <mark>MPI_</mark>	Applic	Application	<mark>Applicati</mark> M	Ap <mark>MPI</mark>	Applic	Applicati	Applicatio	MP <mark>I</mark> Applicat	ior <mark>M</mark> Ap <mark>N</mark>
96 Appl	ication	MPI_Ba	Application						MPI	Applic <mark>MI</mark> Ap	M Applic M	Applicati <mark>M</mark> F		Applic	Applicatior	MApplicati 🗸	1 <mark>P</mark> Ap <mark>MP</mark>	Applicat	Applicati	/FApplicati	MApplicatio	o <mark>MP</mark> Ap <mark>M</mark>
7 Appl	ication		Application						MPI_I	Applica	MApplica N	Applicatio	MAppl <mark>MF</mark>	Applica	Applicatio	Applicatio	MAp MI	Applica	Applicatio	MApplicat	o <mark>M</mark> Applicat	io <mark>M</mark> Appl
8 Appl	ication	MPL B	Application						MP	Applic: <mark>M</mark> App	M Applica <mark></mark> M	1 <mark>Applicatic</mark> M	1P <mark>Ap</mark> MPI	ApplicaM	Application	MApplicatio	MAp №	Applica	Applicatio	MApplicat	ic <mark>MI</mark> Applicati	ior <mark>M</mark> Ap
lqqA 9	ication	MPL B	Application							Applic	1		X			V .	ĂΠ		ľ	1	V	X
210 Appl			Application							ApplicaMFAp	A A	V				<i>(</i> Λ			l I	1	M	
										X	I = I			l W			Χ	. M	1	11	Δ	X
211 Appl	ication	MPI_B	Application						MPI	Applic <mark>MP</mark> Ap	MApplic	IPApplicatio	MAP MPI	Applica	Application	MApplicati M	1PApMF	Applic	Applicatio	Applicati	MApplicatio	o <mark>MP</mark> Ap <mark>M</mark>
12 <mark>Appl</mark>	ication	MPI_Ba	Application						MPI	Applic MAp	I <mark>P</mark> Applic M	I <mark>Applicatic</mark> M	P <mark>Ap</mark> MPI	Applic	Applicatio	M <mark>Applicatio</mark>	M <mark>Ap(MP</mark>	Applic	Applicatio	Applicat	ic <mark>M</mark> Applicati	ic <mark>MP</mark> Ap <mark>№</mark>
913 Appl	ication	MPI_Ba	Application						MPI	Applic MAp	1FApplic	Application		Applic	Applicati <mark>M</mark>	Applicatio	1FAp <mark>MP</mark>	Applic	Applicatio	Applicati	MApplicatio	or <mark>MF</mark> App
P14 Appl	ication	MPI_Ba	Application						MP	Applic <mark>M</mark> Ap(N	<mark>/F</mark> ApplicN	Applicati	Ap <mark>MPI_</mark> E	Applic	Application	n IFApplicati <mark>M</mark>	1Ap1 <mark>MP</mark> 1	Applic	Applicatio	MIApplicat	d Applicati	ioi <mark>M</mark> Ap <mark>M</mark>
P15 Appl	ication	MPI Ba	Application						MP	Applic <mark>M</mark> Ap <mark>M</mark>	P Applic M	Applicati	Ap <mark>MPI B</mark>	Applic	Applicatior	Application	Ap <mark>MPI</mark>	Applic	Applicatio	MApplicati	MApplicatio	or <mark>M</mark> Ap M
	0.5 s	10-	1.5 s	20-	2.5 s	20-	3.5 s	10-	4.5 s			.5 s		6.5 s			.5 s		8	8.5 s		9.5 s
		1.0 s		2.0 s		3.0 s		4.0 s		5.0 s		(6.0 s		7.¢	15		8.0	5		9.0 s	*

Ο

	Runnin	g with TB	B (Inter M	PI Tracer)									_				
	•							1							-	4	10. AL
0.	0 s	0.5 s	1.0 s	1.5 s	2.0 s	2.5 s	3.0 s	3.5 s	4.0	s 4.5	5.0 s		5.5 s	6.0 s	6.5 s	7.0 s	7.5 s
PO	Application			MPApplication	Applicatio	opliApplicatio	MPApplication	MFAppli <mark>MPL</mark>	Applicat <mark>M</mark>	I <mark>I</mark> Application MP	Application MAP	p <mark>MPI_Barri</mark> /	Applicat MF4	Application <mark>MI</mark>	Pl_Sendre <mark>Applicati</mark>	on <mark>MI</mark> Application I	M <mark>Applic MPI_Barrier</mark>
P1	Application		MPI_Ba	rri Application	Applicati	op <mark>/</mark> Applicati	Application		Applicat	Application MP	Application	MPI_Barri	Applicat <mark>MF</mark> A	pplication	Application	MApplication	Pl_Applic <mark>MPl_Barrier</mark>
P2	Application			Application	Applicati <mark>MPL</mark> Ap	p <mark>MI</mark> Applicat	RI_Application	MPApplica	Applicati	4FApplication V	Application MPA	op <mark>MPI_Bar</mark> iA	Applicat <mark>M</mark> A	Application Mi	Application MPI_Se	ndreApplication	MApplicatio MPI_Ba
P3	Application	6.	2	19 Application	Applicatio	p <mark>M</mark> Applicatio	MApplication		Applicat	PApplication M	Application MA	op <mark>MPI_Barr</mark> 4	Applicat MA	Application Mi	PApplication MPI_S	endApplication	A IP <mark>Applicatio</mark> MPI_Ba
Ρ4	Application		MPI_Ba	arr <mark>i</mark> Application	MApplicati <mark>MP</mark> App	M <mark>PI</mark> Applicati <mark>N</mark>	1 <mark>F</mark> Application	1PI_S <mark>(App</mark> //IPI_	Applicat	Application MP	Application MPIA	or <mark>MPI_Barr</mark> 4	Application	pplication/	M Applicati	on <mark>MI</mark> Application	MIApr <mark>MPI_Barrie</mark>
-5	Application		MPI_Ba	arriApplication	MApplicat	MPApplicat <mark>M</mark>	PI_Application		Applica	Application MF	ApplicationMA	op <mark>MPI_Barr</mark> 4	Application	pplication <mark>MF</mark> I	Application	MApplication	MPI_SApplic
P6	Application		MPI Ba	arriApplication	MApplicati MPI A	op <mark>M</mark> Applicati	Application	IPI_S¢Appli MP	Applicat	Application	Application MPI	Ap <mark>MPI Ba</mark> 4	Applicat	Application M	Application MPI_S	endreApplication	MPI Application MPI
P7	Application			MP Application	Applicatio	plicatio	MPApplication		Applicati	MF Application	Application Mi	App <mark>MPI B</mark> a	Applicat 1	Application M	IFApplication MPI	SendrApplication	MFApplicatior MPL 1
P8	Application		MPL B	anApplication			-				Application MPI		<u>uu</u>			ion <mark>MF</mark> Application	MIApr <mark>MPI Barri</mark>
P9				arrApplication			$1 \rightarrow$				PI ,Application Mi					V	MPI SeiApplicati MPI
	Application		MPI Ba		MApplicati MPApp		X			ll A	Application MPLA					onMPApplication	MPIAprMPI_Barri
								Λ I	Ŭ,	N	A X		J.U.	У	A	A	X
	Application		MPI_Ba						Į.	A\ /	MPApplication MPA		114	1		ion <mark>MI</mark> Application	MPApp <mark>MPI_Barr</mark>
	Application		MPI_Ba	arriApplication			I i	X	1	W V	Application <mark>MPI</mark> Ap					ion <mark>MPI</mark> Applicatio	$\overline{\mathbf{X}}$
P13	Application		MPI_Ba	arriApplication	MApplica MPI App	MP <mark>I</mark> ApplicaM	Application	MiApr <mark>MPI_Barr</mark>	Applica <mark>M</mark>	l'Application MPI	Application MIAp	MPI_Barrie	Applica <mark>MI</mark> A	pplication <mark>MP</mark>	ApplicationMPI_Se	endrecApplication	MP Applicat
P14	Application		MPI_Ba	arriApplication	MApplicat MPApp	MP Applicative	Application	App <mark>MPI_Barri</mark>	Applica	PApplication MP	Application <mark>MPI</mark> Ap	plication	Applicat <mark>MP</mark> A	Application MF	Pl_SendreApplicati	on MApplicatio	MPI Application
	Application	61		arri Application		MPI_Applicativ		App <mark>MPI_Barri</mark>			Application	-	Applicat <mark>MP</mark>		MI <mark>Applicat</mark>	ion MApplication	M <mark>App<mark>MP1_Barrie</mark></mark>
0.	0 s	0.5 s	1.0 s	1.5 s	2.0 s	2.5 s	3.0 s	3.5 s	4.0	s 4.5	5.0 s		5.5 s	6.0 s	6.5 s	7.0 s	7.5 s 🗸



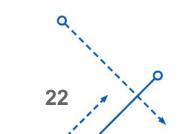
	Slurm.sh	F	Readme
1 2	#!/bin/ <i>bash</i>	1 2	### Bitonic sort
3 4 5	# SBATCHmem=64000	3 4 5	./build.sh
6 7 8	# SBATCH ——exclusive # SBATCH ——constraint=IB	6 7	mpirun -np 8 ./bin/bitonic 1024
9 10	# SBATCHjob-name="702"	8 9	<pre>mpirun -np 8 <tau_exec> ./bin/bitonic 1024</tau_exec></pre>
11 12 13	# SBATCHqos=general-compute	10 11 12	<pre>mpirun -trace ./bin/bitonic 1024</pre>
14 15 16	# SBATCHoutput=~/panasas/logs/bitonic_%x_%j.stdout # SBATCHerror=~/panasas/logs/bitonic_%x_%i.stderr	13 14	sbatch slurm.sh
17 18 19	# SBATCHtime=00:10:00	15 16 17	– install tau pdt to instrument and profile
20 21	# SBATCHnodes=16	18 19	``bash
22 23 24 25	<pre>module load intel-oneapi-2021.3 module load intel-oneapi-mpi/2021.3.0 export I MPI PMI LIBRARY=(usr(lib64/libpmi so</pre>	20 21 22 23	<pre>export TAU_TRACE=1; export TAU_PROFILE=1; export TAU_COMM_MATRIX=1; export TRACEDIR=./tau_trace; export PROFILEDIR=./tau_trace;</pre>
26 27 28 29	<pre>module load intel-tbb/2019.3 source /util/academic/intel/19.3/compilers_and_libraries/linux/tbb/bin/tbbvars</pre>	24	tau_treemerge.pl; tau2slog2 tau.trc tau.edf -o tau.slog2;
30 31 32	<pre>srunmpi=pmi2 /user/zsayed/projects/bitonic-sort/bin/bitonic 1000000000</pre>	27 28 29	jumpshot tau.slog2
33 34	<pre>#mpirun -trace /user/zsayed/projects/bitonic-sort/bin/bitonic 100000000</pre>	30	***

21

Repo: https://gitlab.com/zain_s/bitonic-sort

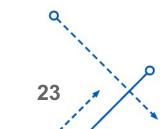
References

- Introduction to Parallel Computing Solutions Manual on the Web Grama, Gupta, Karypis & Kumar
- R. Miller and L. Boxer, <u>Algorithms Sequential and Parallel: A Unified</u> <u>Approach</u>, Third Edition, Cengage Learning, Boston, Mass., 2013.





Thank you!



 \mathbf{n}