Sandbox

Ben Johnson Weitang Liu Agnieszka Łupińska
Muhammad Osama John D. Owens Yuechao Pan
Leyuan Wang Xiaoyun Wang Carl Yang

UC Davis

Contents

Chapter 1

Sandbox for playing with pandoc/slate

Normal table

Operation	Self	Peer	Host	All
Regular read	448.59	14.01	444.74	12.17
Regular write	442.98	16.21	16.18	12.17
Regular update	248.80	11.71	0.0028	6.00
Random read	6.78	1.43	2.39	4.04
Random write	6.63	1.14	3.47E-5	3.82
Random update	3.44	0.83	1.92E-5	2.08

Pretty wide table

Parts	Comp. cost	Comm. cost	Comp. to comm. ratio	Scalability	Memory us
Modularity optimization	10(E + V) /p	20V bytes	E/p : 2V	Okay	88E/p + 1
Graph contraction	5E / p + E'	8E' bytes	5E/p + E' : 8E'	Hard	16E' bytes
Louvain	10(E + V) / p	20V bytes	E/p : 2V	Okay	88E/p + 1

JDO hacked version of above

Parts	Comp cost	Comm cost	Comp/comm ratio	Scalability	Memory usage (B)
Modularity optim.	10(E + V) /p	20V bytes	E/p : 2V	Okay	88E/p + 12V
Graph contraction	5E / p + E'	8E' bytes	5E/p + E' : 8E'	Hard	16E'
Louvain	10(E + V) / p	20V bytes	E/p : 2V	Okay	88E/p + 12V + 16E'

and with math!

Parts	Comp cost	Comm cost	Comp/comm ratio	Scalability	Memory usage (B)
Modularity optim.	$10(E+V)/p$	$20V$ bytes	$E/p : 2V$	Okay	$88E/p + 12V$
Graph contraction	$5E/p + E'$	$8E'$ bytes	$5E/p + E' : 8E'$	Hard	$16E'$
Louvain	$10(E+V)/p$	$20V$ bytes	$E/p : 2V$	Okay	$88E/p + 12V + 16E'$

Even wider table

Parts	Comp. cost	Comm. cost	Comp. to comm
Wedge generation	dE/p		
Wedge communication	0	aE/p x 12 bytes	
Wedge checking	aE/p x log(d)		
AllReduce	2V	2V x 4 bytes	
Triangle Counting	(d + a x log(d))E/p + 2V	aE/p x 12 + 8V bytes	~(d + a x log(
Scan Statistics (wedge checks)	(d + a x log(d))E/p + 2V + V/p	12aE/p + 8V bytes	~ (d + a x log(
Scan Statistics (intersection)	Vdd + V/p	8V bytes	dd : 8

JDO hacked version of above

Parts	Comp. cost	Comm. cost (B)	Comp/comm ratio	Scalability
Wedge generation	dE/p			
Wedge communication	0	aE/p x 12		
Wedge checking	aE/p x log(d)			
AllReduce	2V	2V x 4		
Triangle Counting	(d + a x log(d))E/p + 2V	aE/p x 12 + 8V	~(d + a x log(d)) : 12a	Okay
Scan Statistics (with wedge checks)	(d + a x log(d))E/p + 2V + V/p	12aE/p + 8V	~(d + a x log(d)) : 12a	Okay
Scan Statistics (with intersection)	Vdd + V/p	8V	dd : 8	Perfect

Table with line breaks

Application	Computation to communication ratio	Sc
Louvain	E/p : 2V	Ol
Graph SAGE	~ CF : min(C, 2p)x4	G
Random walk	Duplicated graph: infinity Distributed graph: 1 : 24	Pe
Graph search: Uniform	1 : 24	ve
Graph search: Greedy	Straight forward: d : 24 Pre-visit: 1:24	Pc
Graph search: Stochastic greedy	Straight forward: d : 24 Pre-visit: log(d) : 24	Pc
Geo location	Explicit movement: 25E/p : 4V UVM or peer access: 25 : 1	Ol
Vertex nomination	E : 8V x min(d, p)	Ol
Scan statistics	Duplicated graph: infinity Distributed graph: ~ (d + a * log(d)) : 12	Pe

Application	Computation to communication ratio	Sc
Sparse fused lasso	~ a:8	Le
Graph projection	Duplicated graph : infinity Distributed graph : dE/p + E' : 6E'	Pe
Local graph clustering	(6 + d)/p : 4	Go
Seeded graph matching		
Application classification		

JDO hacked version of above

Application	Computation to communication ratio	Scalability	Implementation diff.
Louvain	E/p : 2V	Okay	Hard
Graph SAGE	~ CF : min(C, 2p)x4	Good	Easy
Random walk	Duplicated graph: infinity	Perfect	Trivial
Random walk	Distrib. graph: 1 : 24	Very poor	Easy
Graph search: Uniform	1 : 24	Very poor	Easy
Graph search: Greedy	Straightforward: d : 24	Poor	Easy
Graph search: Greedy	Pre-visit: 1:24	Very poor	Easy
G.S.: Stochastic greedy	Straightforward: d : 24	Poor	Easy
G.S.: Stochastic greedy	Pre-visit: log(d) : 24	Very poor	Easy
Geolocation	Explicit movement: 25E/p : 4V	Okay	Easy
Geolocation	UVM or peer access: 25 : 1	Good	Easy
Vertex nomination	E : 8V x min(d, p)	Okay	Easy
Scan statistics	Duplicated graph: infinity	Perfect	Trivial
Scan statistics	Distrib. graph: ~ (d + a * log(d)) : 12	Okay	Easy
Sparse fused lasso	~ a:8	Less than okay	Hard
Graph projection	Duplicated graph : infinity	Perfect	Easy
Graph projection	Distrib. graph : dE/p + E' : 6E'	Okay	Easy
Local graph clustering	(6 + d)/p : 4	Good	Easy

Really wide table

Parts	Comp. cost	C
Feature duplication		
Children selection	BC	8I
Child-centric comp.	BCF x (2 + L + Wf1.y + Wa1.y)	4I
Source-centric comp.	B x (CF + (Wf1.y + Wa1.y) x (C + F + Wf2.y + Wa2.y)	0
Graph SAGE	B x (C + 3CF + 3LCF + (Wf1.y + Wa1.y) x (CF + C + F + Wf2.y + Wa2.y))	8I
Direct feature access		
Child-centric comp.	BCF x (2 + L + Wf1.y + Wa1.y)	4I
Graph SAGE	B x (C + 3CF + 3LCF + (Wf1.y + Wa1.y) x (CF + C + F + Wf2.y + Wa2.y))	8I

Parts	Comp. cost	C
Feature in UVM		
Child-centric comp.	BCF x (2 + L + Wf1.y + Wa1.y)	4I
Graph SAGE	B x (C + 3CF + 3LCF + (Wf1.y + Wa1.y) x (CF + C + F + Wf2.y + Wa2.y))	8I

JDO hacked version of above

Parts	Comp. cost
Feature duplication	
Children selection	BC
Child-centric comp.	BCF x (2 + L + Wf1.y + Wa1.y)
Source-centric comp.	B x (CF + (Wf1.y + Wa1.y) x (C + F + Wf2.y + Wa2.y)
Graph SAGE	B x (C + 3CF + 3LCF + (Wf1.y + Wa1.y) x (CF + C + F + Wf2.y + Wa2.y))
Direct feature access	
Child-centric comp.	BCF x (2 + L + Wf1.y + Wa1.y)
Graph SAGE	B x (C + 3CF + 3LCF + (Wf1.y + Wa1.y) x (CF + C + F + Wf2.y + Wa2.y))
Feature in UVM	
Child-centric comp.	BCF x (2 + L + Wf1.y + Wa1.y)
Graph SAGE	B x (C + 3CF + 3LCF + (Wf1.y + Wa1.y) x (CF + C + F + Wf2.y + Wa2.y))

Parts	Comm. cost
Feature duplication	
Children selection	8BC bytes
Child-centric comp.	4B x (F + Wf1.y + Wa1.y) x min(C, 2p) bytes
Source-centric comp.	0 bytes
Graph SAGE	8BC + 4B x (F + Wf1.y + Wa1.y) x min(C, 2p) bytes
Direct feature access	
Child-centric comp.	4B x ((F + Wf1.y + Wa1.y) x min(C, 2p) + CLF) bytes
Graph SAGE	8BC + 4B x (F + Wf1.y + Wa1.y) x min(C, 2p) + 4BCFL bytes
Feature in UVM	
Child-centric comp.	4B x (F + Wf1.y + Wa1.y) x min(C, 2p) bytes over GPU-GPU + 4BCLF bytes over GPU-CPU
Graph SAGE	8BC + 4B x (F + Wf1.y + Wa1.y) x min(C, 2p) bytes over GPU-GPU + 4BCFL bytes over GPU-CPU

Parts	Comp/comm ratio	Scalability
Feature duplication		
Children selection	1 : 8	Poor
Child-centric comp.	~ CF : min(C, 2p) x 4	Good
Source-centric comp.	N.A.	N.A.
Graph SAGE	at least ~ CF : min(C, 2p) x 4	Good
Direct feature access		
Child-centric comp.	~ (2 + L + Wf1.y + Wa1.y) : 4L	poor
Graph SAGE	~ (2 + L + Wf1.y + Wa1.y) : 4L	poor
Feature in UVM		
Child-centric comp.	~ (2 + L + Wf1.y + Wa1.y) : 4L over GPU-CPU	very poor
Graph SAGE	~ (2 + L + Wf1.y + Wa1.y) : 4L over GPU-CPU	very poor