# Single Source, Shortest Path

#### **Kevin Nuss**

#### **CS 530 Parallel Computing**

## **Statistics and Verification**

**Every run saves its runtime and message count** 

Hand check a few small solutions from sequential program

Save solutions from all sequential runs

**Every parallel run checks itself against sequential solutions** 

A few mismatches Too many messages? Only occurred in 400k messages or more Repeat runs did not have the mismatches Program bug?

# **Moore's Algorithm**

The problem involves finding the shortest path from a single, designated source to each of the other vertices in a directed graph that has weighted edges.

Begin with every vertex in a work queue. Try to find a shorter path to another vertex than a direct path from the source. If a shorter path is found, add that destination vertex to the work queue, if not already there. Continue processing vertices that are in the queue looking for shorter paths to other vertices than those already found. If any shorter ones are found, add those destination vertices to the work queue, and so on.

Dijkstra's algorithm uses a similar strategy, but takes advantage of a priority queue which allows each vertex's edges to be examined only once.

### **Termination Strategy** Dual Pass Ring Termination Algorithm

When finished with its work, a designated initiator passes a 'white' token to the next process. When that next process finishes its work, it passes the token onward. However, if a process sends work to another process that is before it in the ring, it changes the next token it receives to 'black.' A process can not know whether the work it sent was to a process that already sent on a 'white' token. Therefore, when a "white" token does arrive, it does not know whether it is still valid.

If the initiator receives a "black" token, it initiates a new white token for another pass.

If the initiator receives a white token, all work is complete because all processes passed on the token without change.

# **Methods of Optimization**

**Consolidate outgoing messages to reduce communication bottlenecks.** 

Receive all available information before calculating shortest paths to reduce wasted effort and wasted outgoing work messages.

Include locally known shortest paths with work messages and tokens to reduce wasted effort and wasted outgoing messages.

Use a priority queue within a process to improve useage of cpu.

### **Centralized Work Queue, by the Book**

X xovm						- <b>G</b> ×
		>	КРУМ 1.2.5 (РУМ 3.4.4) [TID=	0x40001]		
Status: Zooming Done.						
Help:						
File	Hosts	Tasks	Views	Options	Reset	Help
<b>\$</b>		••		••		Time: 5.142877
Trace File: //tmp/xpvm.trace.k	unuss					◆ PlayBack  ◇ OverWrite
			Space-Time: Tasks vs. Tin	ne		
ws00:master						
ws01:slave						and the second sec
ws02:slave						
ws03:slave						
ws04:slave						
ws05:slave						
ws06:slave				- アイン 経営管轄		
ws07:slave						
ws08:slave					「橋子」 フロート・ 「欄」 「安か門」」	
ws09:slave					<b>撒日教</b> 語 注:	
ws10:slave						
ws11:slave	j 🗖 📕 🥬	and the second			1668日:今後第日日日日 1993年をした登場1月1日日 1993年をよう登場1月1日日	
ws12:slave						
ws13:slave						
ws14:slave						
ws15:slave				<u>                                     </u>	<i>XX/WW</i> // // <i>WW</i> //	
vs16:slave						
View Info:						
Computing		User Defined	Overhead	Waiting		Message —

## **Centralized Work Queue** First Try: Combine Messages from Slaves

××	(xpvm X P V M 1.2.5 (PVM 3.4.4) [TID=0x40004]							
St	tatus: Zooming Done.		^	түм 1.2.3 (гүм 3.4.4) [ПО-	0,40004j			
He	elp:							
	File	Hosts	Tasks	Views	Options	Reset	Help	
	\$		•		<b>PP</b>		Time: 0.285102	
Tra	ace File: /tmp/xpvm.trace.kr	nuss				V Pla	vBack 🔶 OverWrite	
				Space-Time: Tasks vs. Tim	le			
	ws00:master							
	ws01:slave			i i i i i i i i i i i i i i i i i i i	i ala kutu kutu kutu kutu kutu kutu kutu kut	- N. Main (N. 1967) (N. 1967)	hi an thi an <b>t</b>	
	ws02:slave					11 - 1000 - 111 - 100 - 2011 - 2010 Galde - 1000 - 2010 - 2010 - 2010 - 2010	TATE THE ALL ALL ALL ALL ALL ALL ALL ALL ALL AL	
	ws03:slave							
	we04:slave							
	ws04.slave							
	ws05:slave							
	ws06:slave							
	ws07:slave							
	ws08:slave							
	ws09:slave							
	ws10:slave							
	ws11:slave							
	ws12:slave							
	ws13:slave							
	ws14:slave							
-	ws15:slave							
	ws16:slave							
d	lose 🔆 🛧 40							
V	iew Info:							
_	Computing		User Defined	Overhead	Waiting 🖂	Messag	e — •	

## **Centralized Work Queue**

#### **Second Try: Priority Queue & Multiple Vertices Sent to Slave in Each Message**

🗙 хрут						
Status: Zooming Done.		×	P V M 1.2.5 (PVM 3.4.4) [TID=0»	(40001]		
Help:						
File	Hosts	Tasks	Views	Options	Reset	Help
3	•	L 🔳		Þ		Time: 0.238419
Trace File: /tmp/xpvm.trace.	knuss					♦ PlayBack ♦ OverWrite
			Space-Time: Tasks vs. Time			
ws00:master						
wsut:slave	╹╹╹╴┼╹╴┼╢╢╢╢┧┧┧	<i>+          </i>		∕ \\/ •   }		
ws02:slave						
ws03:slave						
ws04:slave						
ws05:slave						
ws06:slave						
ws07:slave						
ws08:slave						
ws09:slave						
ws10:slave						
ws11:slave						
ws12'slave						
Wate slave						
ws13:slave						
ws14:slave						
ws15:slave						
VSTB:SIAVE						
Close * 40						
View Info:						
Computing		User Defined	Overhead	1	Waiting 👘	Message —

### **Distributed Queue, by the Book**

X xpvm						
		ХР	V M 1.2.5 (PVM 3.4.4) [TID=0>	(40001]		
Status: Trace Playback Con	nplete					
Help:						
File	Hosts	Tasks	Views	Options	Reset	Help
3			D D	••		Time: 0.411203
Trace File: /tmp/xpvm.trace.k	nuss					PlayBack 🔷 OverWrite
			Space-Time: Tasks vs. Time			
ws00:master						
ws01:slave						
we02*slave						
				an an an taon a Taon an taon an t		
ws05:slave					n an an Albert an Alb Albert an Albert an A	
ws04:slave					8 <b>%</b> MP	
ws05:slave						
ws06:slave					「「「「」」の「「」」の「「」」の「「」」の「「」」の「「」」の「「」」の「	
ws07:slave						
ws08:slave						
ws09:slave		l fr V			· · · · · · · · · · · · · · · · · · ·	
ws10:slave		ŧ.			「「「「「「「」」」	
ws11:slave						
ws12'slave						
		н Ж				
ws13:slave						
ws14:slave						
ws15:slave		N. M. S. S. S. S.				
ws16:slave		1 (1)(7 <b>72</b> 7 (1) (1)(1)				
Close $\frac{*}{*}$ $\frac{40}{*}$						
View Info:						
Computing	Us	ser Defined 📕	Overhead 🗾	Wa	iting 👘	Message —

#### Distributed Queue First Try: Combine Messages and Send Known Minimums

🗙 xpvm					
		X P V M 1.2.5 (PVM 3.4.4) [TID	=0×40001]		
Status: Zooming Done.					
File	Hosts Tasks	Views	Options	Reset	Help
<u>ع</u> اد	•• •		<b>&gt;&gt;</b>		Time: 0.234002
Trace File: /tmn/xnym.trace.k	muss				
		Space Time: Tasks vs. Ti	20		
		opace-time. Tasks vs. th			
ws00:master					
ws01:slave					
ws02:slave				1	
ws03:slave				1	
ws04:slave				1	
ws05:slave					
ws06:slave					
ws07:slave					
ws08:slave				1 1 1	
ws09:slave				+ + +	
ws10:slave					
ws11:slave					
ws12:slave				<u> </u>	
ws13:slave					
ws14:slave					
ws15:slave					
ws16:slave					
Close 💥 🛨 40	И				
View Info:					
, Computing	User Defined	Overhead	Wait	ing 🕅	Message —

## **Distributed Queue** Second Try: Add Priority Queue

🗙 хрут						• 6 ×
		×	РVM 1.2.5 (РVM 3.4.4) [TID=0	x40001]		
Status: Zooming Done.						
Help:						
File	Hosts	Tasks	Views	Options	Reset	Help
3		•	ID D	<b>FF</b>		Time: 0.162367
Trace File: /tmp/xpvm.trace.l	knuss				٦	> PlayBack 🔷 OverWrite
			Space-Time: Tasks vs. Tim	•		
ws00:master						
ws01:slave						
wsuz:siave						
ws03:slave						
ws04:slave						
ws05:slave					/ / //	<u>↓ ╢╴╹ ┪</u> ╹
ws06:slave						
	- / WHAT					
wsu7.slave						
ws08:slave						
ws09:slave						
	XXXX		S & M & M // 1			
ws10:slave						
ws11:slave		NUMBER AND AND AND A			j j	
we12:slave						
Watersate						
ws13:slave						
ws14:slave						
		SECOND X V				
ws16:slave						
						KI
View Info:						
Computing		User Defined	Overhead 🗾	Wait	ing 🦳 Me	essage — 🗧

🗙 xpvm						- 6 ×
		×	P V M 1.2.5 (PVM 3.4.4) [TID=0	x40003]		
Status: Zooming Done.						
Help:						
File	Hosts	Tasks	Views	Options	Reset	Help
3	••			<b>PP</b>		Time: 4.782735
Trace File: /tmp/xpvm.trace.	knuss					🔶 PlayBack 🛛 🗸 OverWrite
			Space-Time: Tasks vs. Time			
ws00:master	1					
we01:elave						
					e shanya shaqaya anni ku	
ws02:slave					A MARINA MARINA MANANANA MANANANA MANANANA MANANA MANANA MANANA MANANA MANANA MANANA MANANA MANANA MANANA MANA	
well3:elavo					·温祉》在1996年1997年1997年1997年1997年1997年1997年1997年	
W305.314VC						
ws04:slave		ALVA N THE AVE	un al american survives dans	ni i n <mark>zv</mark> ili i ni	a <b>Na B</b> ana an Anna an A	
					定期的 建铝石油 有能多可	
wsup:slave			化黄煤间 出口 常識 化二十元间	新生产的 · · · · · · · · · · · · · · · · · · ·	网络古拉福基六十十月二十	
ws06:slave						
					鐵品 搬賣 植隐鸟	
ws07:slave		ANN STATIST	《· A.标题》:《新译》》:《· A.L. 标题 《新译》》:《· 法书》:《· 法书》:《· 》:《· 》:《· 》:《· 》:《· 》:《· 》:《· 》:《·	法在公司撤销股份。 1914年 第二前日撤销股份公司指行撤行	<b>教堂的能法,出版</b> 的公式	
ws08:slave			云凤 明显的 生活力 机能管管子 化合金管 发展制度的 计空间 机能放大力 计合相算		·凯克兰 加加尔兰 [10] [2] ( 	
			MURE CHI MER SUB	的过去分词 化化学 化化学 化化学 化化学 化化学 化化学 化化学 化化学 化化学 化化	· 體行: 職給 行体品並得	<u>An an an</u>
ws09:slave			湖北船 出现的 网络小银行 一地位 人名哈尔福 湖道德 1976年代的 化特尔基 经分配 人名德		创新 植脂 高路 出来	
ws10:slave			14. 当中主义后权公司公司任职工作资源 14. 当中主人的公司 - 中国王王王 - 第18		識 医雄铁己酸白色	
	194.			南上紀爛新 創造都主義	語「読み」「「聞」」語	
ws11:slave			1997年1月1日,1997年4月1日日日(1997年1月1日)(1997年1月) 1998年1日:日本市政部内主任(1997年1月)(1997年5月)	n an the second state of the se The second state of the second st		
ws12:slave					#常心能 /第一位:140、油小。 把一一位相称 / 144版:144版	
				電磁 關於 編結 二編	城口 旅班后 涂印 医白肉	
ws13:slave			NAME AND A CONTRACT OF A CONTRACT			
ws14:slave					AND A CONTRACTOR OF A DATA OF A	
		Xie Xiel				
ws15:slave						
ws16:slave			ANNAL HINNAL ANNA	n jahing jahingan j	ARTSAL ARTALINA ANN ARTAINN	
X						
Close 🜟 🛨 40	М					
View Info:						
Computing		User Defined	Overhead	Waiti	ng 🥅	Message —

#### **Distributed Queue, Second Try - 10k Vertices** Processing of Vertices Commences Before All Processes Have Their Data

🗙 хрут						- 6 ×
		×	P V M 1.2.5 (PVM 3.4.4) [TID=	0x40001]		
Status: Task Height Adjuste	ed.					
Help:						
File	Hosts	Tasks	Views	Options	Reset	Help
<u>~</u>	44	í <b>–</b> I				Time: 18 199951
	<b>``</b>			<u> </u>		
Trace File: //tmp/xpvm.trace.k	nuss				R	> PlayBack   OverWrite
			Space-Time: Tasks vs. Tim	ie		
ws01:slave					<u> 1970 - La Brander La Brand</u>	
ws02:stave						
					II N U DI NII VII VII VII VII VII VII VII VII VI	
ws03:slave						
ws04:slave				┱┰┰┲┅┅╝╾╴╱┅╖╶╸╱╌		Na in the state of
ws05:slave						
ws06:slave	╶╴┼┽╌╌┼┛┼╌╌┼┛╢╢					
					印刷: 1 位的 就的 一口	
ws07:slave					目顎 医内麻痹的 日日	
ws08:slave		<u>╃╺┎╷╢╢┟┼╶╸╢╻╷</u>			물통은 사람을 가을 수 있	
					國新日本 通過 机计 一定	
ws09:slave					에너 요구한 문제가 ?	stantin i 🛄 🕅
ws10:slave		▋▋▋▋▋▋▋		「報告日報」 彩	主張 이렇는 방안 환자 가 가	
				出版的 偏衡 编辑	いたい 長って あたし セ	
ws11:slave			i i i i i i i i i i i i i i i i i i i		《集封》记:"服金"之后	
ws12:slave			TERRET AL CELEMAN AND AND AND AND AND AND AND AND AND A		といわれる しとう装作 うれる	
			438部山口修生)。 載止	14 A 14 H 19 A 2 Y 4	白色 动脉 化化化物	an ta an an till 📶 👘 🖓
ws13:slave					化化学验验机器的 网络小贝科	
ws14:slave			Y E MARTEN AND AND AN		·····································	
				经输出 计算机转移的	a 新聞 自然 新聞 新聞 新聞 新聞 新聞	
ws15:slave			L NAVEL A CALENDARIA A CALENDARIA E kanada a calendaria a calendari			
ws16:slave		A A THIRD AND A THE A				
A						
Close <u>*</u> 🛨 40	М					
View Info:						
Computing		User Defined	Overhead	N N	Vaiting Me	essage —

### All Versions of Single Source, Shortest Path



### **Explanation of Previous Graph**

"Sequential Moore's" is the sequential version of Moore's algorithm. It does not use a priority queue.

"Central by the book" is a naive implementation of the parallel pseudocode provided by the textbook.

"Central, First Try" is an initial attempt to improve the textbook's algorithm by combining messages from the slaves to the master.

"Central, Second Try" adds a priority queue to the master and allows it to send more than one vertex to a slave at a time.

"Central with Data Time" includes the time required to distribute the edge data to the slave processes. Each slave gets all edge data.

"Distrib with Data Time" is my best implementation of the distributed work queue but includes the time to distribute the data to the slaves. Work by a slave begins as soon as that slave has its partial set of edge data.

"Distrib, Second Try" is my best implementation of the distributed work queue version of the program. It includes a priority queue within each process.

"Distrib, First Try" does not have a priority queue. Like the second try, each process receives a subset of the edge data and when it discovers a new shorter path, it sends a message to the appropriate process that has that subset. The master process creates the data, distributes it, manages the termination strategy, records runtime, and checks solution.

"Distrib by the Book" is a naive approach to implementing the parallel pseudocode provided by the textbook.

"Sequential" is a sequential implementation of Dijkstra's Algorithm for finding single source, shortest path. It is the best known method.

## **Possible Improvements**

**MPI Versions of Fastest Algorithms** 

**Better Queue Handling** 

Improve Memory Management Release message buffers immediately Send large data in smaller pieces so message does not compete for memory.

**Improve Centralized Strategy** 

Only send part of edge data to slaves to reduce setup time and increase capacity before virtual memory is needed. Perhaps track which processes already have which edges and manage work queue accordingly.