CSE 138 Lecture 17 this time: - online systems vs. offline systems, row data vs. derived data - intro to MapReduce - MapReduce example: inverted index from forward index ("services") or ("online systems") > systems that accept requests from clients, and respond to clients. - KVSes - web scruers - caches - databases (sometimes) these all wait for client requests and try to handle them, ideally quickly. -low lateracy (time between request and response) is important - high availability (every request receives a response) is important ->" batch processing systems" aka "offline systems" - Take in a huge amount of data at once - Process it (might take hours/days)
- Produce output data
(also potentially huge) - High throughput is important throughput=how much data ran we process in X arount of time?) -> users accessing Facebook data about how people use Facebook derived data & (raw data) new data result of comes in and some transformation goes into or processing storage Map Reduce is a tool for computing derived data. example: inverted index Web crawler gets raw data: words Document the, quick, brown, fox,... docl docz the, dog, is, cute,.. i, love, my, dog cloc 3 This is formatted in a way that's bad for queries like "give me all documents containing 'fax' " word Documents } fox quick docl doc1 doc2, doc3, doc1 doc Z To compute the inverted index from the forward index: for each document D in forward index: for each word w in D: emit (w, D) for each unique w, combine Ds in a list Lauick, doc1> equick, [doc1]> < Pox, doc1> <fox, [doc1]> 4 dog, doc 1 > <dog, [doc1,doc2,doc3]> <cure, [doc2]> 4 dog, doc 2> < cute, doc 2> and there's our inverted index! This algorithm is simple, but doing at large scale is a hard distrys problem. idea of mapReduce: Abstract away the hord bistributed systems stuff and let people plug in the.
Conceptually simple code
for particular batch
processing tasks. or sometimes quite sophisticated! upu have one part of how to proceed? "the", "quick", "brown", "fox" docl Recall that this is our first step: in forward index: (for each document D A for each word w in D emit (w, D) This can be done in parallel for all the documents! "embarrassingly parallel"

map phase shuffle phat reduce phat (doc), (the quick, brown, ...)>, Lithe, cloc 12 canick, doc12) the, doc1> function (brown, doct >) (dog, doc1) intermediate Kay-value pairs (doc2, (my, dog, is, cute)> <1 ove , doc3 > { cmy, doc 2> Lawick, doc1>5 intermediate K-V pairs M3) (doc3, (i, lak, my, dog) 27 d TX7 master ci, doc35 clove, doc 3> 6dog, doc3> reduce function internediate combine ter poirs pairs w/ same key (hash(key) mod R < dog, (docz, doc 3) <love, (doc3, doc5)> <quick, (doc1)> this data a distributed file system. mverted index! such as GFS. For each word w in D - map function! emit <w, D> combine Os in a list for each unique w) < reduce function! upu could write this
as a Haskell one-liner
with fold, blu !! in Map Reduce you only specify the above two functions that are specific to upur computation, and maybe some configuration settings. The framework does the rest. open - source clone of MapReduce: Hadoop!
and HDFS. its well-known successor is Spark! Many batch processing tasks can fit into this model - distributed grep - search
- distributed sort - distributed word count

								•	0				_									
3	Pl	na:	ses	0	+	۵		10	K	-	1 _o	6	•									
	·M /	3 P	P	ha	و ۲				7 • •	2-						. C	,,	d		4		
	,,,,	^ Y	Y		_			+ ~	\	7	ę	SV VL	γ·\ Λ /	L.	~) (٨	(C	7		
																				ta		
									on	do	. (e	S	<i>j</i> 1	nte	201	ne	di	at	e	,	
								1						S								
										Y	70	2	し	Do		Se.		S				
										ľ	N	۱,۱	-	N	12		M	3				
										•		ن			_	י נ		رد		1		
				-1						1	محال م		~	. ا	!	<u> </u>	1	۔ م	١,	10.0	,,,,,	,
	Sh	ہوں	Fle	PI	70	r2,	ح				(TE	<i>.</i> ,	M	ial I	d	re T	tD		V	po	ربرع)
										7	σ_{c}	red	lu	æ	عم ا	υZ))	Ke	<u>^</u> S	,		
												TA	٤,		Ī	2	7					
												<u>_</u>		J	<u>'</u>	12	IJ					
-	rec	du	æ	P	ho	XS-	e	_		P	n	Z	ra	mi	n	بمرو		ŞϤ	PF	lie	:d	
																		ÝΟ.			• .	
										•	V.	11.	2	0	ハ		in.	e	M	ed	iat	
											'	· • • • • • • • • • • • • • • • • • • •	F	ש	7		٥	n		_		
												7 4	au	ce		W	01	N	<u>~</u>	6	•	
																_						
												۷ ۵	ko	3	6	عملا	لِي	/\				
												20	de	2		10	<u> </u>		•			
5	up	pa	se i	WC W	4		MT.	rt				~(عد	1	٧.		3	>			
	10	~]	11			4	. ! .	را د	L			•	6	- 1 -	1	\					
			1,	(1	74	-)	4	~ (, Z	ے تو	,W	71) •	73))) -	_					
	20	שע	. 2,	(0	NL	5,	86	9	,	ıs	_ ('U	re	-/	>							
			:																			
			ـ لد ر		1					~												
			دطه	ī									M	aq	>	P	he	ZSe	e,			
			< c	ur	۲,	l	>						E	W	ui t		t	e	s'e	2		
			28	og.	,		>)			P	a	,r	5					
						`																
				10.		~	05	~	\			1	5		. ^	. لد		_	_ 4			
		+(dog	/	ر •	س س				~ `		X	, y	ت مم	a l	4 C	R	PV	a.	se,	
			×,	CW	e,	J), د	אא	ے و	ان ر	ز ر				ا <i>و</i> ۵	F	PL	uz	Z LA A	5u	m	
				7		_)			Ur	1	PN	/ I	A		
				•								•			U	LN	191	ue	k	نور	4.	
																					J	