Computer Science @ RIT
B. Thomas Golisano College of Computing & Information Sciences

Parallel Data Mining

Team 2 – Flash Coders
Team Research Investigation
Presentation 3

Foundations of Parallel Computing
Dec 2014

Agenda

- Overview & Computational Problem
- Sequential Program
- Parallel Program
- Demo
- Strong Scaling
- Weak Scaling
- Future Work
- Lesson learned

Overview of Topic

Computational Problem

Gradient descent can take a long time to converge with many training examples, and when it does it may not have found the global optimum

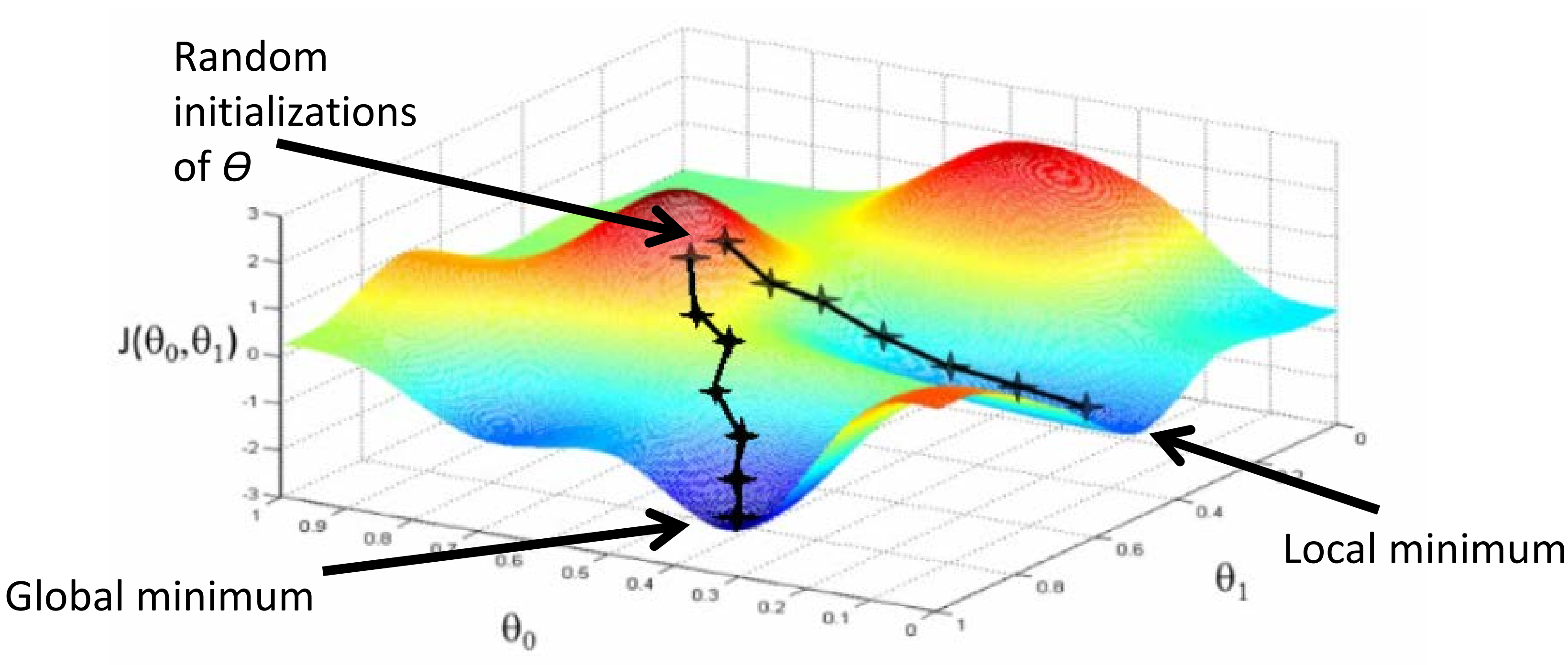


Figure from Vasilis Vryniotis. http://blog.datumbox.com/tuning-the-learning-rate-in-gradient-descent/. Accessed Sept 14, 2014.

Sequential Algorithm I

What if m equals 500 million training examples!

- Linear/logistic regression cost function:

$$\min_{\theta} J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^i) - y^i\right)^2$$

- Algorithm
  - Randomly initialize θ
  - Repeat until convergence (for all j=0,1,...,n):

$$\theta_j = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^i) - y^i\right) {x_j}^i$$

This term is $\frac{d}{d\theta_j} J(\theta)$

Sequential Algorithm II

- Predicted output is then $h_\theta(x^i)$ with trained parameters θ
  - Linear Regression:

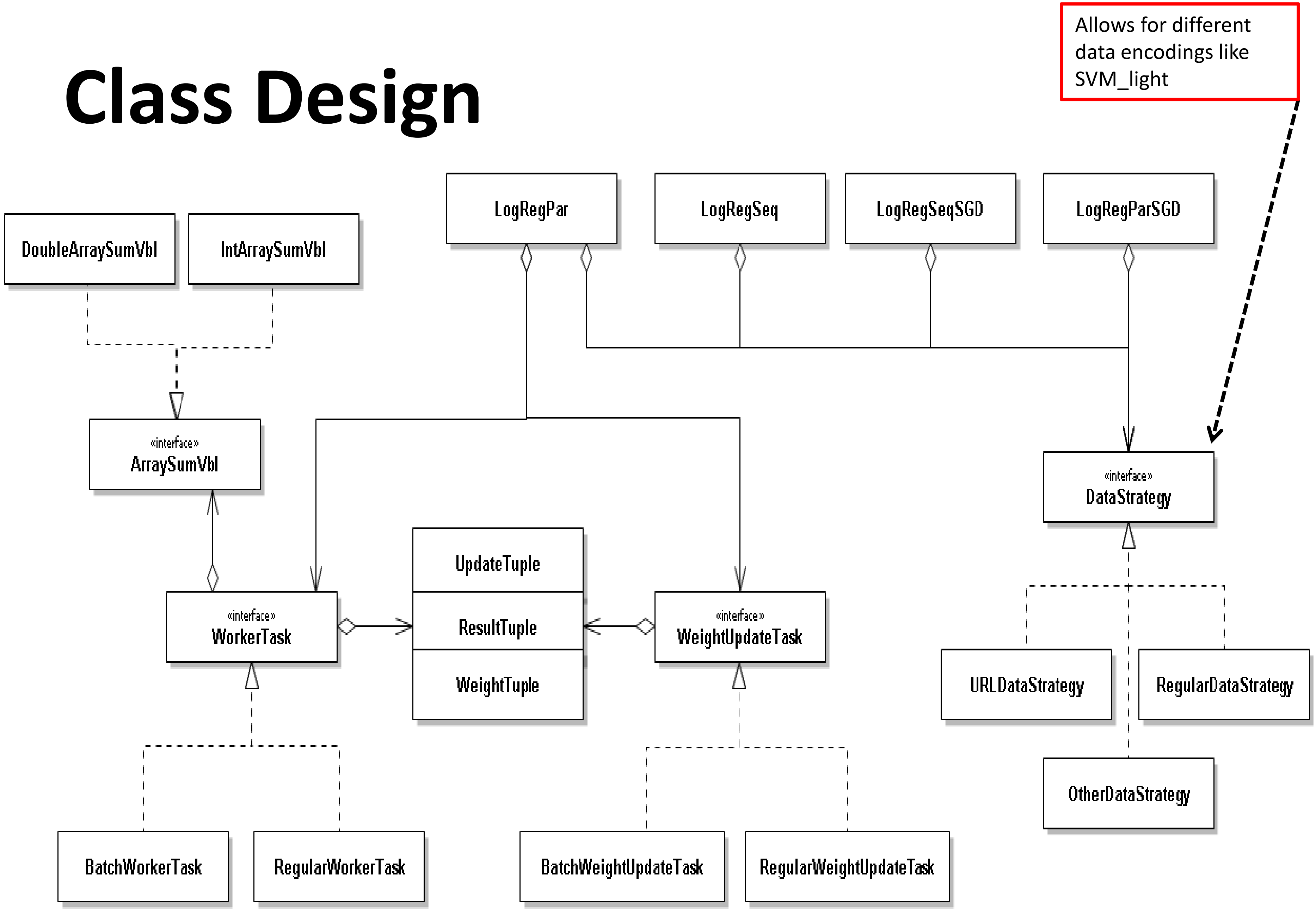
$$h_\theta\left(x^i\right) = \sum_{j=1}^{n} \theta^i x^i$$

  - Logistic Regression

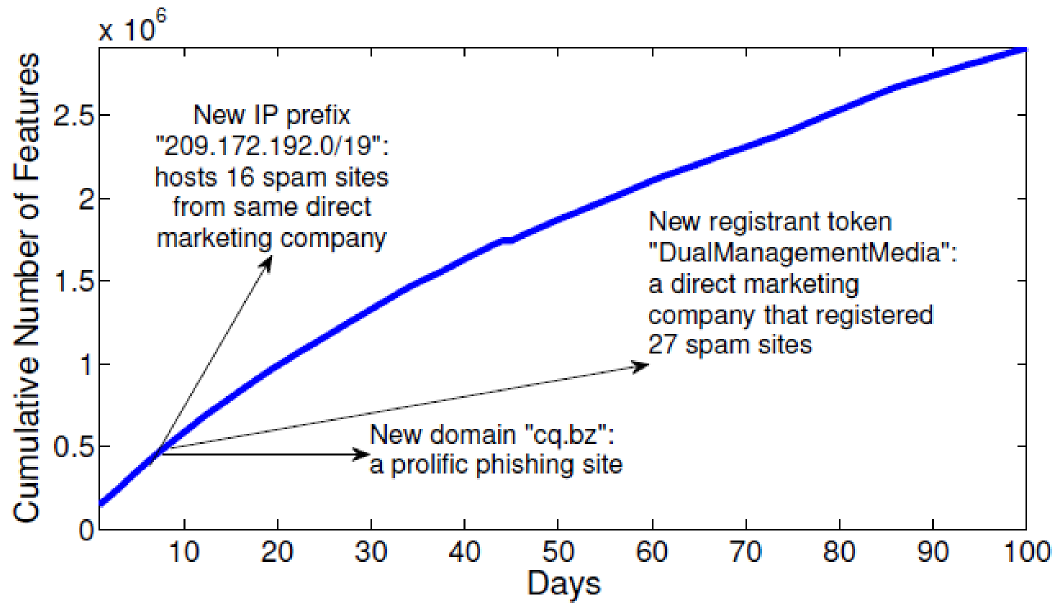
$$z = \sum_{j=1}^{n} \theta^i x^i$$

$$h_\theta\left(x^i\right) = {}^{1}/_{(1+\ e^{-z})}$$

Class Design



Big Data – Suspicious URLs



Instances: 2396130

Attributes: 3231961

Example Instance (anonymized data):

+1 4:0.0414938 5:0.0689655 6:0.176471 26:1 54:1 56:1 62:1 64:1 66:1 68:1 70:1 72:1 933:1 4346:1 65800:1 155153:1 155154:1 155155:1 155166:1 155170:1 155174:1 155175:1 155176:1 155177:1 155178:1 155179:1 155180:1 155181:1 155182:1 155183:1 158035:1 159867:1 1284698:1 1288308:1 2112946:1 2134415:1

Sequential & Parallel Program

Demo

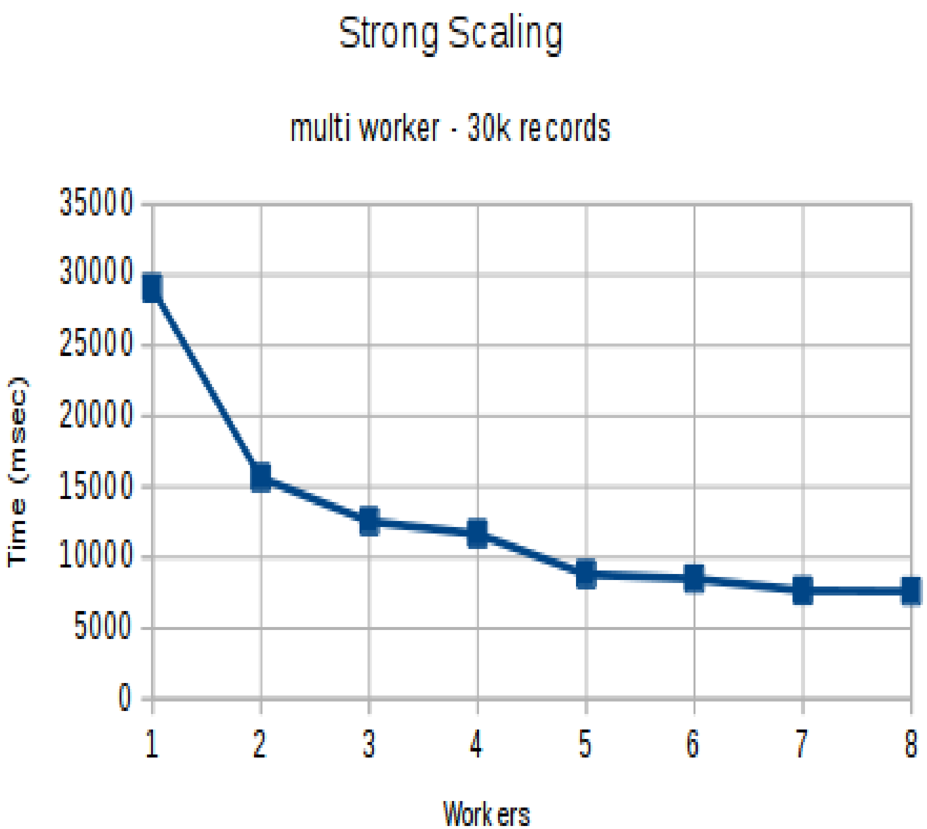
Strong Scaling

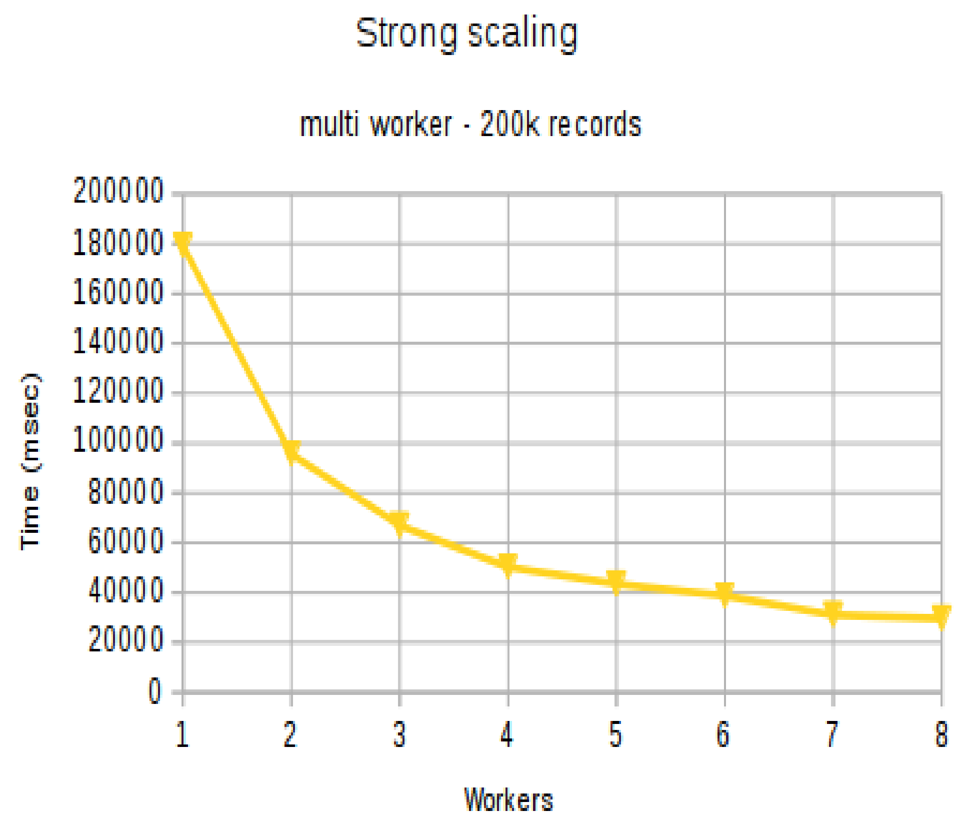
Strong Scaling on Cluster Nodes

Records	Workers	T (msec)	Accuracy	Speedup	Efficiency
30,000	Seq	28972	94.278		
	1	29039	94.278	0.9977	0.9977
	2	15642	94.086	1.8522	0.9261
	3	12547	92.100	2.3091	0.7697
	4	11655	94.031	2.4858	0.6215
	5	8803	93.211	3.2912	0.6582
	6	8494	93.472	3.4109	0.5685
	7	7645	93.887	3.7897	0.5414
	8	7611	92.925	3.8066	0.4758
50,000	Seq	48882	93.507		
	1	49058	93.507	0.9964	0.9964
	2	26406	93.650	1.8512	0.9256
	3	18282	92.293	2.6738	0.8913
	4	14811	93.982	3.3004	0.8251
	5	13950	93.644	3.5041	0.7008
	6	12954	93.450	3.7735	0.6289
	7	7545	93.344	6.4787	0.9255
	8	9755	93.170	5.0110	0.6264

Records	Workers	T (msec)	Accuracy	Speedup	Efficiency
75,000	Seq	75692	91.827		
	1	75746	91.827	0.9993	0.9993
	2	37761	92.621	2.0045	1.0023
	3	26769	93.152	2.8276	0.9425
	4	23610	93.874	3.2059	0.8015
	5	16816	93.111	4.5012	0.9002
	6	15370	93.308	4.9247	0.8208
	7	16654	93.717	4.5450	0.6493
	8	15105	93.079	5.0111	0.6264
100,000	Seq	88290	93.345		
	1	88037	93.345	1.0029	1.0029
	2	51384	92.962	1.7182	0.8591
	3	35328	93.294	2.4992	0.8331
	4	30375	93.834	2.9067	0.7267
	5	27204	93.348	3.2455	0.6491
	6	19429	93.065	4.5442	0.7574
	7	18598	93.729	4.7473	0.6782
	8	16565	93.337	5.3299	0.6662
200,000	Seq	179828	92.799		
	1	179243	92.799	1.0033	1.0033
	2	95628	92.674	1.8805	0.9402
	3	66983	91.938	2.6847	0.8949
	4	50253	93.322	3.5785	0.8946
	5	43535	92.802	4.1307	0.8261
	6	38787	93.318	4.6363	0.7727
	7	31036	93.158	5.7942	0.8277
	8	29632	93.516	6.0687	0.7586

Strong Scaling on Cluster Nodes





Strong Scaling Results

- As the number of workers increases, the problem size for each worker node (N/W) decreases but internode communication remains constant
- This decreases the ratio of computation to communication – not ideal for strong scaling
- BUT: strong scaling allows us to perform regression on datasets that wouldn't otherwise fit in memory – 1 node of tardis (~8GB memory) dies on full dataset but 10 nodes handle it easily

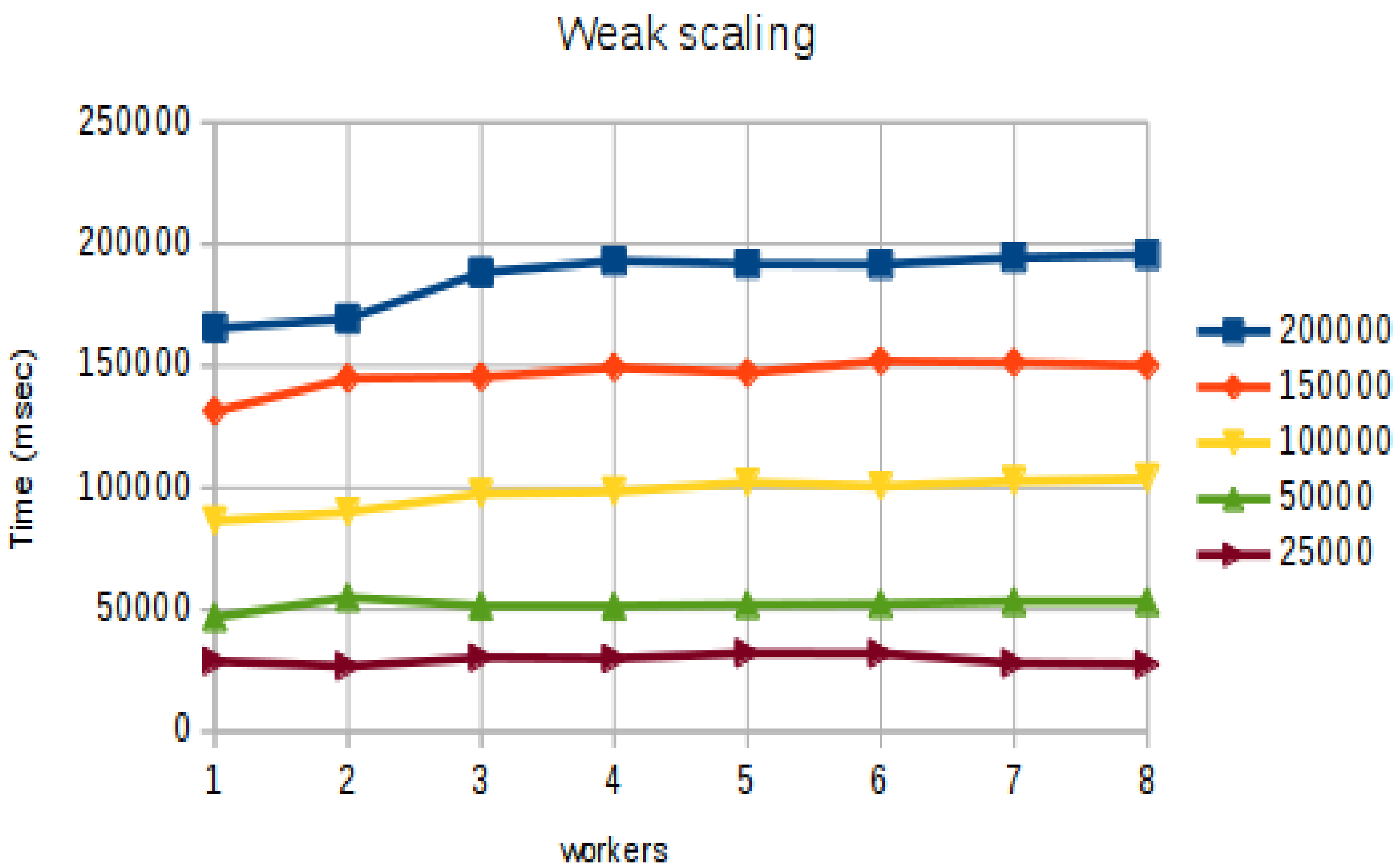
Weak Scaling

Weak Scaling on Cluster Nodes

Records	Records	Workers	T (msec)	Accuracy	Sizeup	Efficiency
25,000		Seq	28568	93.387		
	x1	1	28906	93.387	0.9883	0.9883
	x2	2	26793	93.419	2.1577	1.0789
	x3	3	30348	93.584	2.6486	0.8829
	x4	4	29952	93.403	4.0529	1.0132
	x5	5	32246	93.275	4.6443	0.9289
	x6	6	32058	93.466	6.0352	1.0059
	x7	7	28062	93.209	7.9968	1.1424
	x8	8	27570	93.261	8.1428	1.0178
50,000		Seq	46793	93.670		
	x1	1	46964	93.670	0.9964	0.9964
	x2	2	55050	92.942	1.7062	0.8531
	x3	3	51582	92.588	3.2017	1.0672
	x4	4	51522	92.881	4.0047	1.0012
	x5	5	52049	93.148	4.9494	0.9899
	x6	6	52391	92.600	5.9608	0.9935
	x7	7	53376	92.976	6.8708	0.9815
	x8	8	53342	92.939	8.0051	1.0006

Records	Records	Workers	T (msec)	Accuracy	Sizeup	Efficiency
100,000		Seq	86291	94.134		
	x1	1	86348	94.134	0.9993	0.9993
	x2	2	89994	93.673	1.9190	0.9595
	x3	3	97665	93.153	2.7644	0.9215
	x4	4	98436	92.629	3.9687	0.9922
	x5	5	102178	92.965	4.8169	0.9634
	x6	6	100589	93.209	6.0948	1.0158
	x7	7	102744	93.298	6.8532	0.9790
	x8	8	103819	93.090	7.9172	0.9896
150,000		Seq	131592	93.334		
	x1	1	131428	93.334	1.0012	1.0012
	x2	2	144903	92.788	1.8140	0.9070
	x3	3	145440	93.355	2.9889	0.9963
	x4	4	149426	92.427	3.8933	0.9733
	x5	5	147138	93.058	5.0778	1.0156
	x6	6	152049	93.043	5.8062	0.9677
	x7	7	151443	93.043	7.0280	1.0040
	x8	8	150240	92.815	8.0641	1.0080
200,000		Seq	165862	93.323		
	x1	1	165546	93.323	1.0019	1.0019
	x2	2	169329	93.830	1.9553	0.9777
	x3	3	188334	93.075	2.6973	0.8991
	x4	4	193105	92.660	3.9012	0.9753
	x5	5	191880	92.949	5.0319	1.0064
	x6	6	191667	92.546	6.0067	1.0011
	x7	7	194444	92.951	6.9000	0.9857
	x8	8	195472	92.807	7.9579	0.9947

Weak Scaling on Cluster Nodes



Weak Scaling Results

- The cluster parallel program does require tight coupling on each iteration
- As opposed to in strong scaling, in weak scaling here the amount of computation to communication remains almost the same
  - The internode communication required increases only slightly by adding another worker
- This leads to good weak scaling results

Current Work: SGD

- Parallel Stochastic Gradient Descent
  - 97.4% accuracy on 2396130 records using all 3231962 features
  - 4496247 msec (~1hr) for 1 million iterations - most data was NEVER seen prior to classification
  - 95.8% accuracy in ~7 minutes (20,000 iterations)
  - Parallelized over weight update step
    - Only significant for millions of features
  - SGD doesn’t use regularization so there is a risk of overfitting the data

Current Work: Mini-Batches

Note: $w = \theta$

$$w_t = \operatorname*{argmin}_{w \in \Omega} \Big[\underbrace{\phi_{I_t}(w)}_{\text{General logistic regression}} + \underbrace{\frac{\gamma_t}{2} \|w - w_{t-1}\|_2^2}_{\text{Batch Penalty – w has larger changes in beginning and smaller at the end}} \Big]$$

- Prevents divergence from previous consensus found
- Limits dramatic changes of parameter w over batch iterations

Current Work: Mini-Batches

- Mini-Batch Gradient Descent on all data in 50 batches / 50 features:

  Logistic regression took 341.475 seconds in 15000 iterations

  Init: Cost 0.6587 Sq Error 0.2124

  Final: Cost 0.1744 Sq Error 0.0455

  Number of features with bias term: 51

  Classification took 0.035 seconds

	-1	1
-1	301667	19184
1	8854	150085

  Accuracy: 94.156%

  Precision: 88.667%

  Recall: 94.429%

  94% accuracy in ~6 min on entire dataset

  20% test set

- Can't compare directly with strong/weak scaling – does less iterations to converge
- Three different parameters to optimize – batch size, batch cost, batch iterations
- Found that too small of a batch size leads to poor accuracy
- Further research required

Future Work I

- Tuning parameters - optimal parameters not obvious:
  - Seed: random restarts
  - Alpha: learning rate
  - Epsilon: convergence threshold
  - Steps: number of iterations
  - Threshold: class decision parameter
  - Lambda: regularization parameter
  - Records: amount of data
  - Features: number of features
  - Batched: number of batches
  - Gamma: batch cost
  - Biter: iterations for one batch
  - Test: percent of training examples set aside

Future Work II

- Feature selection to only use the best of the ~3 million features
- Automatic parameter selection using validation set
- Parameters that change over time

Lessons Learned

- Tuple space does not like ~3 million features being transferred twice per iteration
- Ex:
  - 50 features: Logistic regression took 13.346 seconds in 100 iterations (73% accuracy)
  - 3,231,962 features: Logistic regression took 438.815 seconds in 100 iterations (66% accuracy)
  - 3,000% increase
- Version control (should have used it)

References

- Sameer Singh, Jeremy Kubica, Scott Larsen and Daria Sorokina. 2009. Parallel Large Scale Feature Selection for Logistic Regression. In *Proceedings of 2009 Society for Industrial and Applied Mathematics* (SIAM) Data Mining, SIAM, Philadelphia, PA, USA, 1172-1183. **URL**:http://epubs.siam.org/doi/pdf/10.1137/1.9781611972795.100
- Haoruo Peng, Ding Liang, and C. Choi. Evaluating parallel logistic regression models. In *Proceedings of the 2013 IEEE International Conference on Big Data*, pp 119-126, 6-9 Oct 2013. **URL**:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6691743&isnumber=6690588
- Mu Li, Tong Zhang, Yuqiang Chen, and Alexander J. Smola. 2014. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '14). ACM, New York, NY, USA, 661-670. **URL**:http://dl.acm.org/citation.cfm?id=2623330.2623612&coll=DL&dl=ACM&CFID=415399891&CFTOKEN=69514427
- Justin Ma, Lawrence K. Saul, Stefan Savage, and Geoffrey M. Voelker. 2009. Identifying suspicious URLs: an application of large-scale online learning. In *Proceedings of the 26th Annual International Conference on Machine Learning* (ICML '09). ACM, New York, NY, USA, 681-688. **URL**:http://cseweb.ucsd.edu/~savage/papers/ICML09.pdf

Further References

- Michele Banko and Eric Brill. 2001. Scaling to very very large corpora for natural language disambiguation. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics* (ACL '01). Association for Computational Linguistics, Stroudsburg, PA, USA, 26-33. **URL**: http://dl.acm.org/citation.cfm?id=1073017
- Mohammed Javeed Zaki. 1999. Parallel and Distributed Data Mining: An Introduction. In *Revised Papers from Large-Scale Parallel Data Mining, Workshop on Large-Scale Parallel KDD Systems, SIGKDD*, Mohammed Javeed Zaki and Ching-Tien Ho (Eds.). Springer-Verlag, London, UK, UK, 1-23. **URL**: http://dl.acm.org/citation.cfm?id=744383
- Andrew Ng. Machine Learning course materials, Coursera. https://www.coursera.org/course/ml. Accessed September 10, 2014.

Code:

https://github.com/rcmccartney/parallel_regression

Questions