INTRODUCTION TO DISTRIBUTED-PARAMETER NETWORKS pdf

1: CiteSeerX â€” Citation Query Introduction to Distributed Parameter Networks

introduction to distributed parameter pdf Introduction to the geometric distribution. Consider a sequence of trials, where each trial has only two possible outcomes (designated failure and success).

Kahng, Sudhakar Muddu - Proc. European Design Automation Conf , " In performance-driven interconnect design, delay estimators are used to determine both the topology and the layout of good routing trees. We address the class of moment-matching, or moment representation, methods used to simulate interconnects modeled as distributed RC or RLC lines. We provide accurate 2- and 3-segment equivalent circuits for the distributed RLC and distributed RC models. Our equivalent circuits approximate a distributed RLC structure accurately up to second degree terms. We have evaluated our models using the two-pole methodology for voltage response calculations. As routing trees become bigger and interconnection lines become longer, our approach has greater advantages in both accuracy and simulation complexity. Show Context Citation Context In this paper, we develop new segmented interconnect representations for delay simulation in interconnect trees; our goal is to improve both accuracy and efficiency over existing methodologies. Design Automation by Andrew B. Kahng, Sudhakar Muddu - 31st Conference on, 0: The traditional analysis of signal delay in a transmission line begins with a lossless LC representation, which yields a wave equation governing the system response; 2-port parameters are typically derived and the solution is obtained in the transform domain. In this paper, we begin with a distribut In this paper, we begin with a distributed RC line model of the interconnect and analytically solve the resulting di usion equation for the voltage response. A new closed form expression for voltage response is obtained by incorporating appropriate boundary conditions for interconnect delay analysis. The discussion furthermore provides a unifying treatment of the past three decades of RC interconnect delay analyses. Mey [17] noted the crudeness of this approximation and proposed an infinite partial fraction expansion, thus obtaining the same solution as Sakurai. The common feature of all these works is that they use the 2-port transfer matrix of the distributed RC line to obtain their respec Time course of potential spread along a skeletal muscle fiber under voltage clamp by W. Knox Chandler, Martin F. Schneider - Journal of General Physiology , " Best least squares fits of calculated curves to data obt The concept of two lumped-distributed RC band-pass filters with a high Q factor realized in the transistor-only filters domain is presented. Both circuits are based on a differential amplifier with a unity gain and a simple lu Both circuits are based on a differential amplifier with a unity gain and a simple lumped-distributed RC notch filter. The schems are obtained using the complementary transformation. The analysis results are presented. The main advantage is the tunability of the time-constant. Wong, Avant Corporation - Proc. Abstract â€” In this paper, we present a fast and accurate delay estimation method for buffered interconnects. The interconnect wire is modeled by the transmission line model which is more accurate and efficient than lumped circuit model. For the interconnect wire, we specify the wire shape to be of t By using first three poles in the transfer function, we derive analytical expressions for calculating delay at any threshold voltage under a finite ramp input. The expressions involved in calculating coefficients in the transfer function are also analytical. We use k-factor equations to estimate delays for buffers. Since the k-factor equations require a loading capacitance for delay computation, we use the effective capacitance technique introduced in [17] to calculate the effective capacitance for each interconnect wire which is connected to a buffer. Therefore, our delay calculation for buffered interconnects is analytical and thus very efficient. Wong , " In this paper, we consider continuous wire-sizing optimization for delay minimization and ringing control. In this paper, we specify the wire shape to be of the form 99 the Elmore delay model suggest that exponential wire shape is effective for delay minimization. The relevant transmission line equations are solved by using Picard-Carson method. The transient the actual delay delay is minimized, or the wiring area is minimized subject to a delay bound. Our method for delay estimation is very efficient. We also find that in determining the optimal shape which minimizes delay, the Elmore delay model performs as good as the our method in terms of the minimum actual delay it achieves, i. However, in determining the optimal shape which minimizes area subject to a delay bound, the Elmore delay model performs much worse than our method. Eisenberg , " We study the electric

potential and field produced by disordered distributions of charge to see why clumps of charge do not produce large potentials or fields. The question is answered by evaluating the probability distribution of the electric potential and field in a totally disordered system that The question is answered by evaluating the probability distribution of the electric potential and field in a totally disordered system that is overall electroneutral. An infinite system of point charges is called totally disordered if the locations of the points and the values of the charges are random. It is called electroneutral if the mean charge is zero. In one dimension, we show that the electric field is always small, of the order of the field of a single charge, and the spatial variations in potential are what can be produced by a single charge. In two and three dimensions, the electric field in similarly disordered electroneutral systems is usually small, with small variations. Interestingly, in two and three dimensional systems, the electric potential is usually very large, even though the electric field is not: If the system is locally electroneutralâ€" as well as globally electroneutralâ€"the potential is usually small in all dimensions. The properties considered here arise from the superposition of electric fields of quasi-static distributions of charge, as in non-metallic solids or ionic solutions. These properties are found in distributions of charge far from equilibrium. Screening; shielding; disordered charge. Tail Asymptotics Throughout this subsection we assume that a is smal

INTRODUCTION TO DISTRIBUTED-PARAMETER NETWORKS pdf

2: Distributed parameter system - Wikipedia

Enter your mobile number or email address below and we'll send you a link to download the free Kindle App. Then you can start reading Kindle books on your smartphone, tablet, or computer - no Kindle device required.

Distributed Deep Learning, Part 1: An Introduction to Distributed Training of Neural Networks This post is the first of three part series on distributed training of neural networks. Introduction Modern neural network architectures trained on large data sets can obtain impressive performance across a wide variety of domains, from speech and image recognition, to natural language processing to industry-focused applications such as fraud detection and recommendation systems. But training these neural network models is computationally demanding. Although in recent years significant advances have been made in GPU hardware, network architectures and training methods, the fact remains that network training can take an impractically long time on a single machine. Fortunately, we are not restricted to a single machine: In model parallelism, different machines in the distributed system are responsible for the computations in different parts of a single network - for example, each layer in the neural network may be assigned to a different machine. In data parallelism, different machines have a complete copy of the model; each machine simply gets a different portion of the data, and results from each are somehow combined. Of course, these approaches are not mutually exclusive. Consider a cluster of multi-GPU systems. We could use model parallelism model split across GPUs for each machine, and data parallelism between machines. While model parallelism can work well in practice, data parallelism is arguably the preferred approach for distributed systems and has been the focus of more research. For one thing, implementation, fault tolerance and good cluster utilization is easier for data parallelism than for model parallelism. Model parallelism in the context of distributed systems is interesting and does have some benefits such as scalability to large models , but here we will be focusing on data parallelism. Data Parallelism Data parallel approaches to distributed training keep a copy of the entire model on each worker machine, processing different subsets of the training data set on each. Data parallel training approaches all require some method of combining results and synchronizing the model parameters between each worker. A number of different approaches have been discussed in the literature, and the primary differences between approaches are Parameter averaging vs. Parameter Averaging Parameter averaging is the conceptually simplest approach to data parallelism. With parameter averaging, training proceeds as follows: Initialize the network parameters randomly based on the model configuration Distribute a copy of the current parameters to each worker Train each worker on a subset of the data Set the global parameters to the average the parameters from each worker While there is more data to process, go to step 2 Steps 2 through 4 are demonstrated in the image below. In this diagram, W represents the parameters weights, biases in the neural network. Subscripts are used to index the version of the parameters over time, and where necessary for each worker machine. For the mathematically inclined, the proof is as follows. Consider the case of a cluster with n workers, where each worker processes m examples, for a total of nm examples processed between averagings. First, how should we implement averaging? The naive approach is to simply average the parameters after each iteration. While this can work, we are likely to find that the overhead of doing so to be impractically high; network communication and synchronization costs may overwhelm the benefit obtained from the extra machines. Consequently, parameter averaging is generally implemented with an averaging period in terms of number of minibatches per worker greater than 1. However, if we average too infrequently, the local parameters in each worker may diverge too much, resulting in a poor model after averaging. The intuition here is that the average of N different local minima are not guaranteed to be a local minima: What averaging period is too high? Some preliminary research on the subject such as [8] suggests that averaging periods on the order of once in every 10 to 20 minibatches per worker can still perform acceptably well. Model accuracy is of course reduced as the averaging period is increased. An additional complication related to optimization methods such as adagrad, momentum and RMSProp. However, these updaters have internal state typically 1 or 2 state values per network parameter - should we average this state also? Averaging the internal updater state should result in faster convergence in each worker, at the cost of doubling - or more - the total size of the network transfers.

The primary difference between the two is that instead of transferring parameters from the workers to the parameter server, we will transfer the updates i. This gives an update of the form: Architecturally, this looks similar to parameter averaging: Readers familiar with the mathematics of training neural networks may have noticed an immediate similarity here between parameter averaging and the update-based approach. This equivalence also holds for multiple averaging steps and other updaters not just simple SGD. Update-based data parallelism becomes more interesting and arguably more useful when we relax the synchronous update requirement. Async SGD has two main benefits: First, we can potentially gain higher throughput in our distributed system: Second, workers can potentially incorporate information parameter updates from other workers sooner than when using synchronous every N steps updating. These benefits are not without cost, however. By introducing asynchronous updates to the parameter vector, we introduce a new problem, known as the stale gradient problem. The stale gradient problem is quite simple: By the time a worker has finished these calculations and applies the results to the global parameter vector, the parameters may have been updated a number of times. This problem is illustrated in the figure below. A naive implementation of asynchronous SGD can result is very high staleness values for the gradients. For example, Gupta et al. For N executors, this means that the gradients will be on average N steps out of date by the time they are applied to the global parameter vector. This has real-world consequences: Most variants of asynchronous stochastic gradient descent maintain the same basic approach, but apply a variety of strategies to minimize the impact of the stale gradients, whilst attempting to maintaining high cluster utilization. It should be noted that parameter averaging is not subject to the stale gradient problem due to the synchronous nature of the algorithm. Some approaches to dealing with stale gradients include: For example, the system of [4] delays faster workers when necessary, to ensure that the maximum staleness is below some threshold All of these approaches have been shown to improve convergence over the naive asynchronous SGD algorithm. Of note especially are the first two: Soft synchronization [9] is quite simple: Parameters are then updated according to: The combination of softsync and stalenessâ€”dependent scaling performs better than either does alone. Decentralized Asychronous Stochastic Gradient Descent One of the more interesting alternative architectures for performing distributed training of neural net- works was proposed by [7]. This paper is interesting for two primary reasons: No centralized parameter server is present in the system instead, peer to peer communication is used to transmit model updates between workers. Updates are heavily compressed, resulting in the size of network communications being reduced by some 3 orders of magnitude. In a standard data parallel implementation using either parameter averaging or async SGD , the size of the network transfers are equal to the parameter vector size as we are transferring either copies of the parameter vector, or one gradient value per parameter. The residual vector is added to the original update: Put another way, large updates per parameter are dynamically transmitted at a higher rate than small updates. Two questions arise here: The answers are a lot and less than you might expect. Take for example a model with Compression Update Size Reduction None bit floating point As impressive as the results are, there appear to be three main downsides of this approach. Strom reports that convergence can suffer in the early stages of training using fewer compute nodes for a fraction of an epoch seems to help Compression and quantization is not free: Distributed Neural Network Training: Which Approach is Best? So which one should we prefer in practice? For one thing, we could define different approaches as best, according to any of the following criteria: That said, there seem to be some conclusions we can draw from the research: Synchronous parameter averaging or equivalently, synchronous update-based approaches win out in terms of accuracy per epoch, and the overall attainable accuracy, especially with small averaging periods. However, the additional synchronization costs mean that this approach is necessarily slower per epoch; that said, fast network interconnects such as InfiniBand can go a long way to keeping synchronous approaches competitive see for example [5]. Adding compression should further help to reduce network communication overheads. Consequently, synchronous systems are less viable as the total number of workers increases. Asynchronous stochastic gradient descent is a good option for training and has been shown to work well in practice, as long as gradient staleness is appropriately handled. Some implementations such as softsync approach described earlier can be viewed as spanning a continuum between nave asynchronous SGD and synchronous implementations, depending on the hyperparameters used.

INTRODUCTION TO DISTRIBUTED-PARAMETER NETWORKS pdf

Async SGD implementations with a centralized parameter server may introduce a communication bottleneck by comparison, synchronous approaches may utilize tree-reduce or similar algorithms, avoiding some of this communication bottleneck. Utilizing N parameter servers, each handling an equal fraction of the total parameters is a conceptually straightforward solution to this problem. Furthermore, many of the ideas compression, quantization, etc. For distributed training to be worthwhile, we need the computational benefit of the additional machines to outweigh these overheads. Furthermore, setup time i. Consequently, our advice is simple: Network training times can become prohibitive for two reasons: Often, these go hand in hand; in fact, a mismatch between the two large network, small data; small network, lots of data may lead to underfitting or overfitting - both can lead to poor generalization of the final trained model. Model parallelism using multi-GPU systems may also be viable for large networks. Another perspective is to consider the ratio of network transfers to computation. Distributed training tends to be more efficient when the ratio of transfers to computation is low. Upcoming in Part 2: References [1] Kai Chen and Qiang Huo.

INTRODUCTION TO DISTRIBUTED-PARAMETER NETWORKS pdf

3: CiteSeerX â€” Citation Query Introduction to Distributed-Parameter Networks

DOWNLOAD INTRODUCTION TO DISTRIBUTED PARAMETER NETWORKS introduction to distributed parameter pdf Introduction to the geometric distribution. Consider a sequence of trials, where each trial has only two.

The concept of the lumped-distributed RC notch filters in transistor-only filters domain is presented. Independent tuning of real and imaginary parts of conjugate complex zeros in filter transmittance is considered. Unfortunately this concept has not been used in practice because of lack of control of the time-constant of distributed RC line. The active filters with distributed RC lines were discovered again by Tsividis [13], and he has named them transistor-only filters [14]. Optimal termination of transmission lines excluding radiation by Rohini Gupta, Lawrence T. Pillage - in Proc. Design Automation Conf , " Abstract â€” The design automation of high-speed digital system interconnects is a challenging problem which requires controlling reflections from discontinuities and noise due to crosstalk. On-chip interconnect design automation is well defined since the objective is to minimize R and C to minimize d On-chip interconnect design automation is well defined since the objective is to minimize R and C to minimize delay. In contrast, for boards and packaging interconnect, the design objective is much more difficult to specify in terms of a metric. This paper presents a generalized approach for RLC interconnect design automation. A new metric is defined that specifies the optimal design as a function of input signal rise-time, loading conditions on the line, parasitic resistance in the circuit and discontinuities in the interconnect. The approach is based on the recognition of the relation between moments of the responses and critical damping of the circuit. Simple lumped-distributed RC notch and band-pass filters are reminded. Output resistances of bias circuits are working as lumped resistors. Simple CMOS implementations of such filters are proposed and di Simple CMOS implementations of such filters are proposed and discussed. The analysis results are presented. Showalter , " The regularized layered medium equation is proposed as a model of voltage distribution in a medium consisting of alternating thin films of conducting and dielectric materials. This equation is obtained from the layered medium equation by the introduction of a regularizing perturbation that takes This equation is obtained from the layered medium equation by the introduction of a regularizing perturbation that takes account of the resistance at the interface between adjacent conducting and dielectric layers. The regularized equation is an implicit evolution equation which is shown to be well-posed, and an explicit measure is derived for the rate of decay of singularities in the initial data. It is shown that, as the regularizing parameter approaches zero, solutions of the initial-boundary value problem for the regularized equation converge to the corresponding solution of the layered medium equation. This gives a method of calculating the exact rate of decay of singularities in the initial data for the layered medium equation.

4: Introduction to Distributed Training of Neural Networks

introduction to distributed parameter pdf introduction to distributed parameter pdf Introduction to the geometric distribution. Consider a sequence of trials, where each trial has only two possible outcomes.

5: introduction_to_distributed_parameter_networks

Note: Citations are based on reference standards. However, formatting rules can vary widely between applications and fields of interest or study. The specific requirements or preferences of your reviewing publisher, classroom teacher, institution or organization should be applied.

INTRODUCTION TO DISTRIBUTED-PARAMETER NETWORKS pdf

The Alternatives to Gridlock U.S.PRC relations and the / 3. Training and work opportunities in the UK Outlines Highlights for Multicultural Citizenship by Kymlicka, ISBN Selected poetry of Dan Pagis Imagery in T.S. Eliots poetry How to claim state benefits The Step-By-Step Guide to Patio and Container Gardening (Step-By-Step Guide to Creative) Why Im Guessing Youd Rather Live in Paris than Tehran Patriotic holidays The original Fannie Farmer 1896 cook book The scholar gipsy Models of restorative justice Excuse me, but Im first The transcendental deduction (3) Emperor Penguins (Pull Ahead Books) Reels 126-128. Owensboro Concise history of hong kong In the footsteps of champions Measurement: a way to capture observation in mathematics and science Java a beginners guide sixth edition WHAT DO YOU KNOW? VOLUME 2 (And Not So Common Knowledge) Waiting For Agnes Curriculum and Instruction After Desegregation Entrenching an uneven playing field: the multilateral regulation of agriculture Odessas meteor crater Until Love is Enough (Finding Mr. Right) World history 1918 to 1970 Nine bad shots of golf and what to do about them Dogs Dont Wear Sneakers (Stories to Go!) ROMANCE bNFLUENCE: Campaign strategy Surgery of the skull base Smythe Sewn Swirling Peacock Ivory Unlined Managing Price Risk in Agricultural Commodity Markets (Farm Business Management (Textbooks)) The (check)book of love The Lady Herberts gentlewomen Sammy Franks The Frog With Glasses Govind Ballabh Pant Zagatsurvey 2000: Los Angeles So. California Restaurants (Zagatsurvey: Los Angeles/Southern California Re