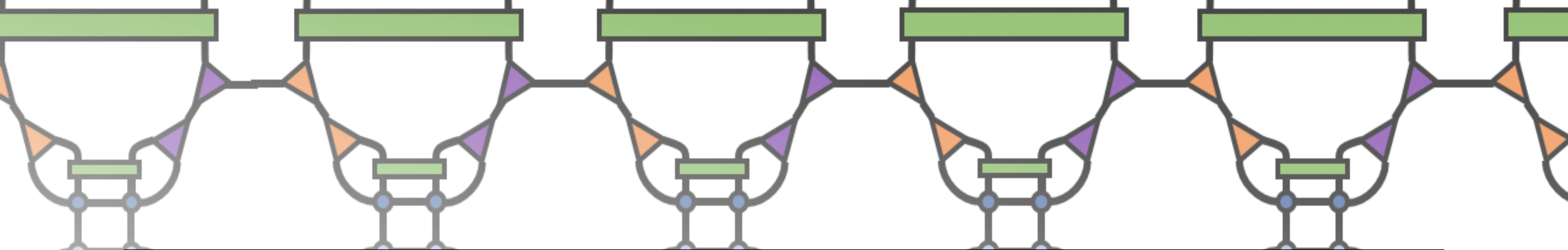


The Basics of Tensor Network

An overview of tensors and renormalization

Samuel Desrosiers, Glen B. Evenbly and Thomas E. Baker

Institut quantique & Département de physique, Université de Sherbrooke, Sherbrooke, Québec, Canada J1K 2R1

Background Image: Decomposition of sites used in the tensor network renormalization (TNR) scheme.

Studying quantum physics has produced some of the most profound discoveries of the past 100 years. Whether determining how a metal breaks, developing new medicines, or making global warming predictions all depend on some level on knowing how electrons interact with other molecules in a given material. Correctly describing how electrons and other particles interact with each other can allow for faster discoveries in the laboratory. Even before developing the next generation of technologies, we can simulate how materials work on a computer. To do this, the equations of quantum physics must be simulated accurately.

Quantum physics is a very abstract study. This is because the answer, known as the wavefunction, has very strange properties. Even though the interpretation of this quantum result can be far beyond the common experience, the answers we obtain can be exceedingly accurate and predict measured phenomena out to several digits in some cases.

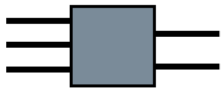
It can very difficult to simulate the core equation of quantum mechanics, Schr̈odinger's equation. One problem that any theoretical physicist faces when using a computer to solve a quantum system is the amount of memory available on a computer. Even if we only want to simulate system of a dozen sites, we would need a supercomputer to store the information!

Clever strategies must be developed to solve quantum systems, or we would lose out on the ability to develop the next generation of technologies. The strategy that we will explore in this document is to use tensor networks. A tensor network takes the large quantum physics problem and breaks it into smaller pieces.

Decomposing the quantum system into tensors requires some effort. In order to make sure that no information is lost in doing this–or that the most relevant information is kept–the amount of information between each tensor must be accounted for.

Tensors and Diagrams

A tensor transforms an object of a certain size and modifies it to be an object of the same or another size. The mathematical vocabulary for tensors are complex even for expert physicists, so a series of diagrams has been created to make tensors more friendly. We can write down a tensor by how many lines come in and how many come out. The following diagram shows an tensor with 3 lines on the left and 2 lines on the right:

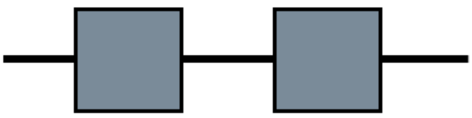


The tensor shown has 5 total lines, and in the mathematical language this is known as a *tensor of rank 5*. One simply counts the number of lines (called indices) on a tensor to find its rank.

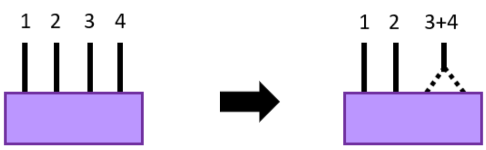
Basic Tensor Operations

Regular numbers like 1 or 2 or 3,000 have operations that can change them. For example, one can add (+) numbers together. Subtraction (-), multiplication (×), or dividing (÷) numbers is also possible. Just like these basic operations, tensors can also be acted upon by a few operations.

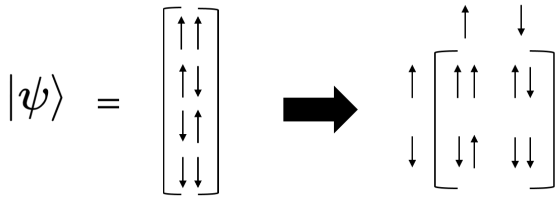
Contraction. Perhaps the most important thing that two different tensors can do is join legs together. For example, these two tensors have been *contracted* along one (each) of their legs.



Reshape. On just one tensor, we can also regroup legs together. For example, we can make reshape two legs to become one:

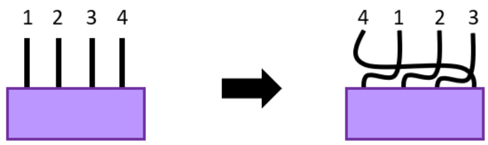


This has a mathematical and physical meaning. If one is familiar with vectors and matrices, then reshaping a vector can give a matrix:

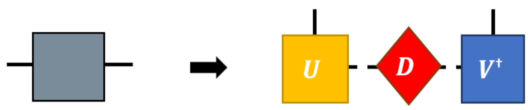


The physical meaning of performing a reshape is that certain quantum states are partitioned. One can think of grouping the quantum states together for an eventual subdivision of the wavefunction.

Permute Dimensions. While crossing two lines of a tensor on the page is an easy operation, a computer requires extra effort to copy the data to a new place. Hence, permuting the order of lines on a tensor requires care when actually implementing a tensor network:



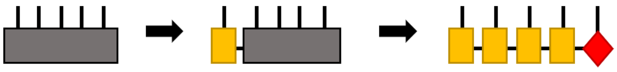
Singular Value Decomposition. While tensors can be joined together by contraction, we can also break apart two tensors with the singular value decomposition (often abbreviated as the SVD). This decomposes a rank 2 tensor into three tensors:



The SVD contains two tensors U (yellow rectangles) and $V^\dagger$ (blue rectangles) that will form new unitary tensors in our system. The third tensor, D (red diamonds), is extremely important. It ranks the most important numbers that are shared between the other tensors. This expresses the entanglement and allows us to compress the information in our network without sacrificing too much accuracy. This can be done by deleting the elements in D with the lowest values.

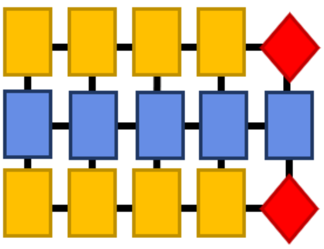
Tensor Networks

With the tools of the last section, we can carry out a series of SVDs to generate a network of tensors that compose a *matrix product state* from a full wavefunction:



That means a wavefunction can be represented as group of tensors, each containing all the information of several sites in the more general form.

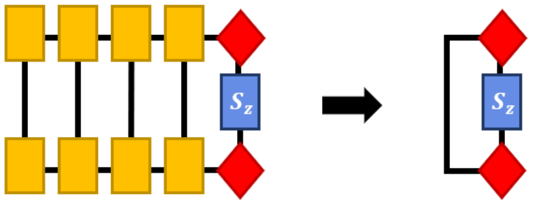
Using these diagrams, an observable quantity (shown is the energy of a quantum system) is drawn as



The blue squares are the Hamiltonian, which contains all the properties of the physical system.

The problem with representing the wavefunction is it is a large rank tensor, which require a lot of memory to store. Tensor network decomposition avoids storing the full wavefunction and therefore can be stored on a regular computer.

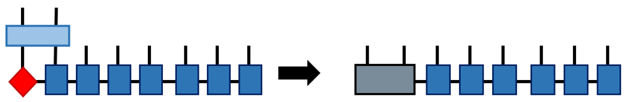
One important class of problems that tensor networks can solve are those that are classified as *locally entangled*. This essentially means that the changes in one tensor only affect a few tensors around it. By careful construction, the red and blue tensors in our diagram contract to a trivial identity tensor. So, for example, if we want to measure the magnetization on one site, we can do it with just three tensors, no matter how many are in the network! This is an example of how tensor networks are useful for finding local properties.



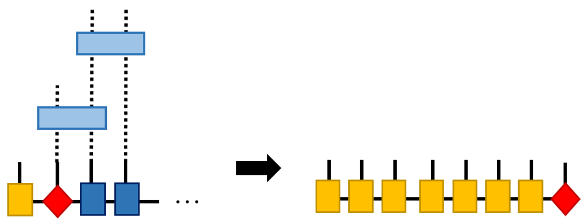
Algorithms

To find the solution to a quantum problem, tensors can undergo the four operations above in many different ways. Each algorithm that uses tensor networks applies the basic operations in different orders and on different tensors.

Time-Evolving Block Decimation To find the solution to a quantum problem, we can use a an algorithm called time-evolving block decimation (TEBD). In TEBD, a time evolution operator can be constructed from the Hamiltonian. By breaking up the full operator, we can apply small two-site gates iteratively along the chain:



Applying that operator many times on our wave function will be driven towards the lowest energy solution for our system. At each step, the resulting tensor obtained by the contraction of the time evolution tensor and the two site tensors (the grey tensor above), is separated again with an SVD.



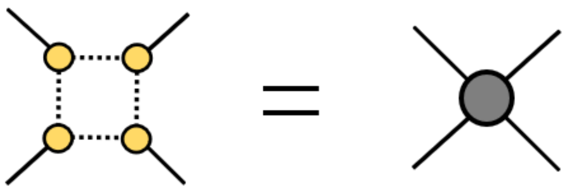
This procedure bring us to the lowest energy state of our system. This same procedure can also evolve our wavefunction in real time with the same steps.

Tensor Renormalization Group There is a way to use tensor networks to study classical systems. Just as in the quantum case, contracting the full network would result in only one high-rank tensor representing our system, which would be too large to store. The general idea behind these algorithms is to represent large systems of particles in interactions with only one tensor without increasing the rank of that tensor.

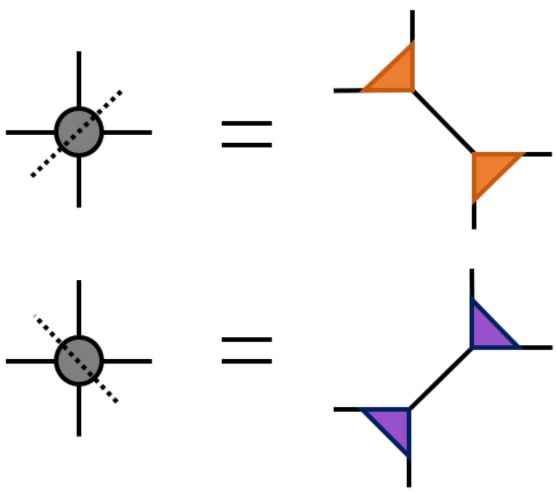
To do this, a renormalization of the problem can be applied. A renormalization of the problem reduces the number of physical sites and also eliminates unimportant degrees of freedom.

The tensor renormalization group (TRG) is an example of how tensor networks can be used to solve these systems efficiently. We form supersized (or coarsened) tensors, while using SVDs to filter the result and obtain an accurate answer. These two steps are repeated until convergence.

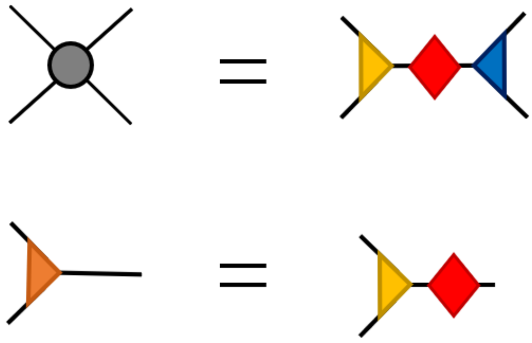
For all these algorithms we begin with an initial tensor representing the partition function–an object from which many useful quantities of the system can be deduced–of some initial state at the lowest scale:



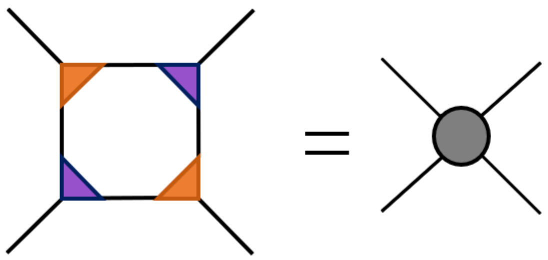
Each index of the tensor accesses a different spin state of the system. The next step is to take a series of SVDs to break up the tensor:



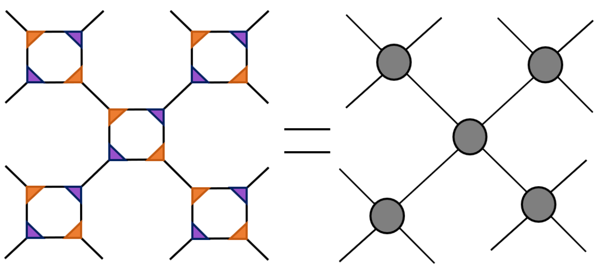
We define new tensors such as:



Tensors from adjacent sites can be joined together:



The resulting tensor is of the same rank as the original while contracting our system and only keeping the important information for solving our problem.



The resulting tensor can then be used again for another TRG step. This operation can be performed on finite or infinite systems.

Many other algorithms (tensor network renormalization, etc.) can solve these two-dimensions classical systems, but the general idea is still the same: represent a large system by a tensor with a low rank while keeping only the most useful information for our problem.

Conclusion

A brief overview of how tensor networks can represent quantum problems was presented in this article. Tensor networks have a lot of applications and it is important to keep developing them so that we can solve quantum problems. The basic operations of a tensor network and two algorithms–time-evolving block decimation and tensor renormalization group–were summarized.

References

[1] U. Schollwöck, *Reviews of Modern Physics* **77**, 259 (2005)

[2] U. Schollwöck, *Annals of Physics* **326**, 96–192 (2011)

[3] G. Vidal, *Physical Review Letters* **91**, 147902 (2003)

[4] M. Levin and C.P. Nave, *Physical Review Letters* **99**, 12601 (2007)

[5] G. Evenbly and G. Vidal, *Physical Review Letters* **115**, 180405 (2015)