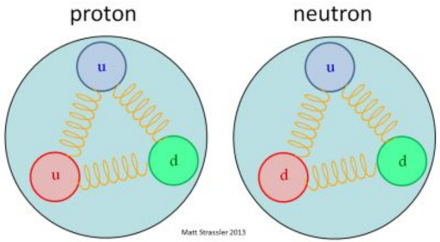
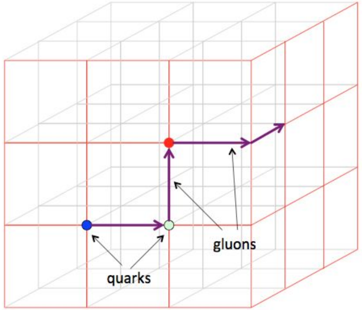
UDEL HACKTHON

5DSpeedsters



- Higg's Boson is the origin of mass in the Universe? WRONG! It's mainly **QCD.**
- QCD, Quantum Chromodynamics, is the theory of the **strong force**; as the name suggests, it's strong, much stronger than electromagnetism, gravity and the weak force.
- The QCD binding energy comprises over 99% of the mass to the proton and neutron (and basically every other composite particle).
- QCD, unlike the other forces, gets **stronger** at larger length scales. At the length scale of hadrons (protons, neutrons, etc) the coupling constant is **too large** for traditional Taylor-expansion based methods to work; i.e. pencil and paper methods don't work!
- Instead we *simulate* the theory on a uniform (Euclidean) space-time grid. Here the quarks live on the sites and the gluons (force exchange particles) live on the links between sites.
- Our simulations are used to understand the **Standard Model** and are compared to experimental results in **new physics** searches.



Our Problem: Lattice QCD

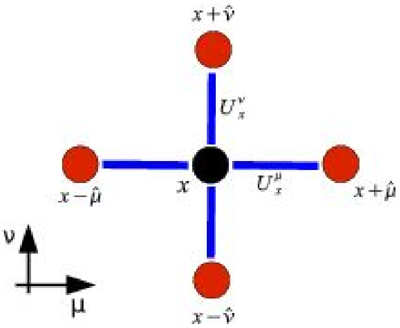
- Simulation comprises 2 parts:

1) Generate *gauge configurations* - snapshots of the quantum vacuum that contains the dynamics of the theory. The function that defines the theory can be interpreted as a probability distribution from which we sample using Monte Carlo techniques.

2) Measurement upon those configurations. Typically these are correlation functions of quantum operators computed as sums over regions of the lattice of products of matrices and vectors.

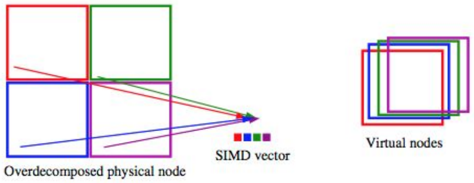
- Our dominant computational expense is computing the matrix inversion of the **Dirac matrix**. This matrix has (4*3*V)^2 complex entries; for V=64^3 x 128, the number of entries is **1.62x10^17** !
- Fortunately it is sparse so iterative localized inversion algorithms like Conjugate Gradient work.
- Fundamental operation is a linearized 4D derivative where in each of the 8 directions we accumulate a 3x3 complex matrix multiplying a 3*4 complex vector.

Typical simulations require months of running on supercomputers. e.g. a 100M BG/Q core hour job corresponds to 6400 rack hours with each rack running at 50 Tflops!



Our Code: Grid github.com/paboyle/Grid

- Grid is a new C++ library developed for Lattice QCD that aggressively uses C++11 features (auto type, lambda, STL)
- Grid employs a SIMD friendly layout:
  - The local lattice per node is divided into subdomains (logical nodes) that map perfectly into the available SIMD length.



- Two levels of parallelism exploited:
  - Vectorization. Currently relies on intrinsics to get the best performance.
  - OpenMP "parallel for" loops to loop over the domain-decomposed sites.
- Performs well on Intel CPUs.

Grid SP Mobius CG performance (2016/1)

	BlueWaters	Edison	CoriP1	Babbage
Cores/node	16	24	32	60
Peak(SP) GF/s	627	921	2335	1000
Bidi Network (GB/s)	9.5	11	11.5	
Single node (Gflops/s)	117	265	630	290
8^4 multinode	29	82	88	
16^4 multinode	43	130	190	

```
template<class Impl>
void WilsonKernels<Impl>::DiracOptDhopSiteDag(StencilImpl &st,DoubledGaugeField &U,
                                              std::vector<SiteHalfSpinor,alignedAllocator<SiteHalfSpinor> >  &buf,
                                              int sF,int sU,const FermionField &in, FermionField &out)
{
 SiteHalfSpinor  tmp;   SiteHalfSpinor  chi;   SiteHalfSpinor *chi_p;
 SiteHalfSpinor Uchi; SiteSpinor result; StencilEntry *SE;  int ptype;
 SE=st.GetEntry(ptype,Xp,sF);
 if (SE->_is_local ) {
   chi_p = &chi;
   if ( SE->_permute ) {
     spProjXp(tmp,in._odata[SE->_offset]);
     permute(chi,tmp,ptype);
   } else {
     spProjXp(chi,in._odata[SE->_offset]);   <<<<<<<<<<<<<<<<<<<< 4x3 complex matrix -> 2x3, local operation
   }
 } else {
   chi_p=&buf[SE->_offset];
 }

 Impl::multLink(Uchi,U._odata[sU],*chi_p,Xp,SE,st);   <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< F(x+mu) = U_mu(x) F(x)
 spReconXp(result,Uchi);
```

Goal(s)

- Long-term goal: To port Grid to GPUs with OpenACC
- This Hackathon: Feasibility study and initial implementation.
    - Move Wilson Dslash Kernel (WilsonKernels<Impl>::DiracOptDhopSiteDag) into Open ACC loop. Typically most performance critical part of the code. Has been working with Mat.
    - Currently failing to compile in one of the routines. (multLink).
    - Compile with -ta=tesla(:managed) -> work with Mat to get PGI 16.5
    - Generate a call graph of Wilson Dslash to track down. (Grid is highly templated and the dependencies are not very transparent)

```
template<class Impl>
 StencilImpl : defines nearest neighbor    DoubleGaugeField : connection between FermionField

 void WilsonFermion<Impl>::DhopInternalCommsThenCompute(StencilImpl & st,DoubledGaugeField & U,
                                                   const FermionField &in, FermionField &out,int dag) {
st.HaloExchange(in,compressor);    <<<<<<<<<<<<<<<<<<< communication with neighoring sites, nodes.
if ( dag == DaggerYes ) {
PARALLEL_FOR_LOOP                 <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< SITE LOOP, can turn into openAC pregma
       for(int sss=0;sss<in._grid->oSites();sss++){
         Kernels::DiracOptDhopSiteDag(st,U,st.comm_buf,sss,sss,in,out);
        }
      }
```