Compiler, Heterogeneous Parallel Computing and Mathematical Optimization

Lanting Guo

September 2, 2015

Background

```
Foucus on machine learning
three mountatians ==> one mountatian
MLer() = compute_ability(HPC) + algorithm(OPT) + model(DL,
iterative, coordinate ascent and active learning:
   iterative: again, again, again, ....
   coordinate ascent: a simple heuristic algorithm, optimiz
   active learning: learn what you like, learn what you are

minimize: the height of three mountatians
subject to: sum(time_i) = 10000hours
            keep other states relatively stable and persist
```

Compiler

`LLVM`
`kaleidoscope`
`Julia`
`Cxx.jl`

```
\begin{figure}
\begin{center}
    \includegraphics[width=4in,height=3in]{images/llvm_tk1
\end{center}
\end{figure}
```

Classical compiler design -1

```
\begin{center}
    \includegraphics[width=4in,height=3in]{images/llvm_tk2
\end{center}
```

Classical compiler design -2

```
\begin{center}
    \includegraphics[width=4in,height=3in]{images/llvm_tk3
\end{center}
```

Julia: 2013_jeff[type, synctax], gc, gf, speed, easy to use

Heterogeneous Parallel Computing

julia: multi-thread, multi-task MC demo

CUDA C: Monto Caro demo

ClusterManagement.jl

Mathematical Optimization

```
solve L1-norm optimization
```

an machine learning example

```
cosmic OR kaggle
```