A WORKLOAD AWARE MODEL OF COMPUTATIONAL RESOURCE SELECTION FOR BIG DATA APPLICATIONS

Amit Gupta, Weijia Xu
{agupta , xwj}@tacc.utexas.edu

Natalia Ruiz-Juri, Kenneth Perrine
nruizjuri@mail.utexas.edu, kperrine@utexas.edu

Texas Advanced Computing Center

Center for Transportation Research

University of Texas at Austin

BIG DATA COMPUTATIONAL LANDSCAPE

- Volume and Variety of Data increasing at a rapid pace
- Analysis Workloads also increase in complexity
- Place increasing demands on Computing Infrastructure
- For maximum performance, vendors respond specialized hardware
  - GPUs, Accelerators (Intel Xeon Phi)
  - Intel Knights Landing many-core

BIG DATA APPLICATION WORKLOADS

- Stress the hardware in different ways
  - IO Bound
  - Computationally Intensive
  - Storage
  - Network Intensive
- Computing Infrastructure (HPC)
  - CPU Architectures
  - Caching layers and algorithms
  - Memory Technologies
  - Storage Technologies
  - Network Interconnects

Problem: **Best Infrastructure** for **given Workload** ?

RELATED RESEARCH AREAS

- Workload Characterization
  - Gaining a granular low level picture of the Application being examined
    - Low level execution traces (Intel PIN tool), Hardware performance counters
  - Usually done with one objective in mind
    - Energy optimization
    - CPU/Resource utlization
- Performance Benchmarking
  - Use an application representative of a class of workloads and compare systems
  - Industry standard benchmarks
    - SPEC benchmarks
    - HiBench, SparkBench

MOTIVATIONS & GOALS

- Workload Characterization
  - We're not trying to optimize the inner workings of an application for an objective
- Performance Benchmarking
  - Translating benchmark numbers to real application performance is somewhat vague
- Performance bottlenecks vary with datasets
  - IO latency becomes apparent when input dataset is large
- Our Goal
  - Predict ***relative performance*** for an application across different available hardware

COMPUTING INFRASTRUCTURE

- Two architectures styles that will be prevalent
  - Multi core
    - A few really fast cores
    - Intel Xeon processors
    - Stampede
  - Many core
    - Several (at-least an order higher in number) of moderately slow cores
    - Intel Knights Landing processors (KNL)

STAMPEDE

- Each Node
  - 2x Intel Xeon E5 Sandy Bridge processors
  - 16x 2.4GHz Hardware Threads in total
  - 32G Memory
- Mellanox FDR Infiniband technology
  - 2 Level (cores and leafs) topology

KNL

- Each Node
  - 1x Intel Xeon Phi 7250 (Knights Landing)
  - 272x 1.4GHz Hardware Threads in total
  - 96GB Memory
    - 16GB is fast MCDRAM technology
- Omnipath 100Gb/s network

OUR METHOD

- Supply and Demand model of application demand

- $F(demand, supply) = \begin{cases} low\ utilization & supply > demand \\ optimal\ utlization & supply = demand \\ low\ efficiency & supply < demand \end{cases}$

OUR METHOD

- Each subsystem
  - $time(\text{demand, supply}) = \begin{cases} minimum & supply \geq demand \\ g(demand\ , supply) & supply < demand \end{cases}$
- Total time (sum of all subsystems)
  - $Total = \sum time(demand, supply) = \sum time(application\ parameters, hardware\ parameters)$

OUR METHOD

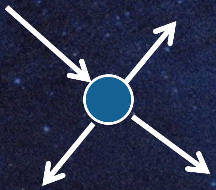
- Using Support Vector Machines on Historical run Data
  - With Appropriate Features carefully selected
    - Hardware Characteristics
    - Parameters of the Application drawn from its Domain Knowledge
  - With enough historical data, achieves high accuracy

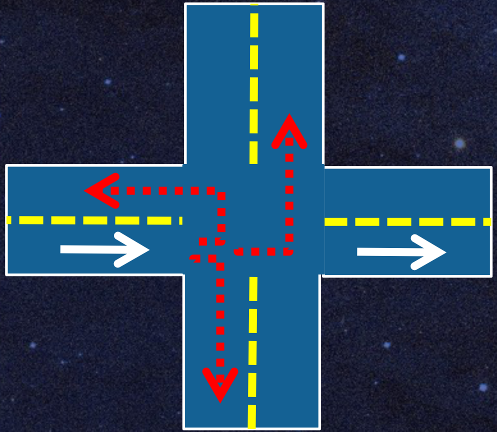
- Infrastructure Provider : Improved Resource Utilization
- End Users : Quicker (Analysis + Experimentation) cycles and lesser application tuning

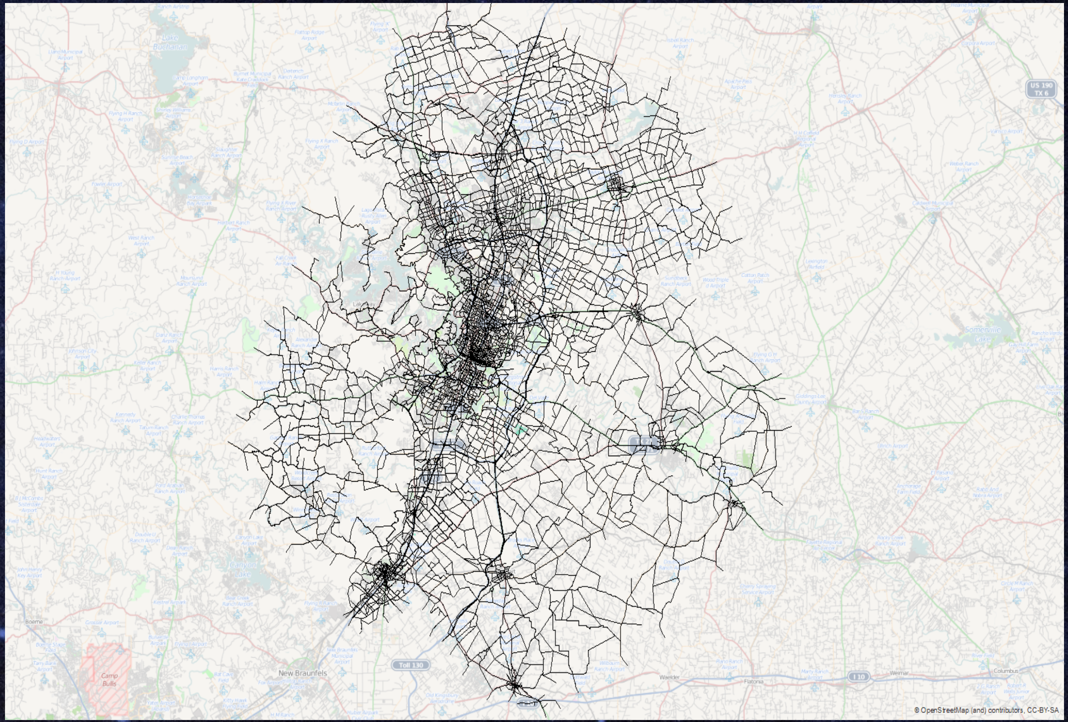
APPLICATIONS (1)

- VISTA
  - Transportation Simulation Framework (C++)
- Dynamic Traffic Assignment (DTA)
  - Models Interactions between
    - Traveller ← → Traveller
    - Traveller ← → Transport Infrastructure
  - Shortest Path computation (Time dependent)
    - Main computational component

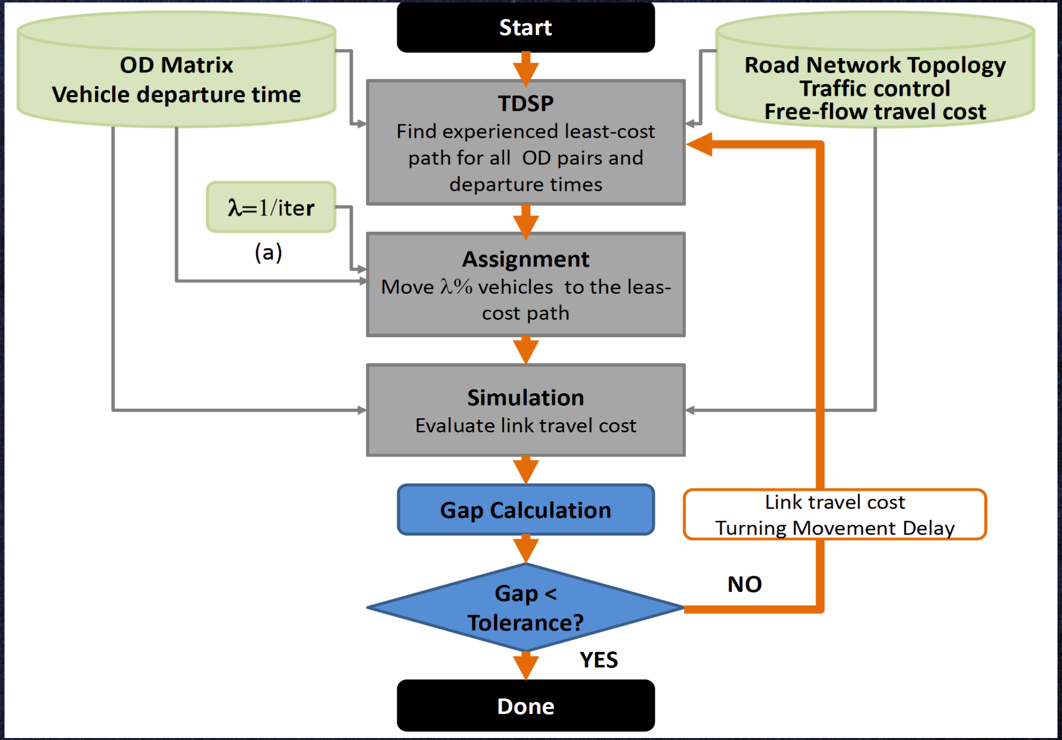
ROAD NETWORKS







DTA WORKFLOW

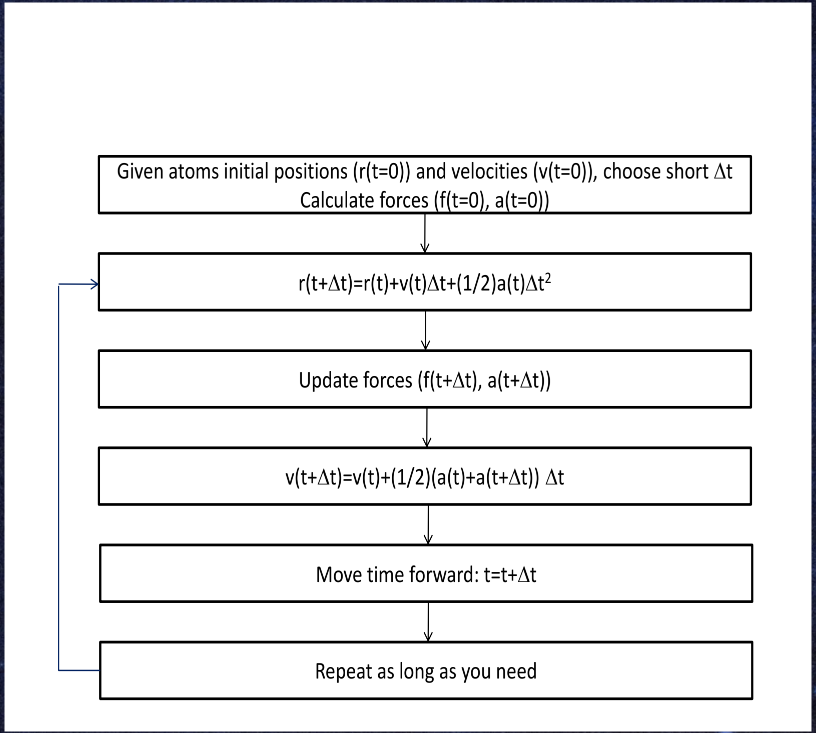


- Iterative
- Graph Based
- Like Big Data problems
  - PageRank
  - Queries on GraphDBs
  - Network Community Detection
- Critical importance societally

APPLICATIONS (2)

- LAMMPS
  - Molecular Dynamics framework (C/C++)
- Simulates interactions between particles in a closed space
- Initial conditions
  - Velocity
  - Force fields
- Other parameters
  - Box dimensions
  - Number of particles
  - Time simulated

LAMMPS WORKFLOW



- Iterative
- System evolves based on Newtons Second Law
- Initial conditions can be set
  - Number of particles/ atoms/molecules
  - Forces
- Simulation Time

SVM

- Support Vector Machines
- Supervised Learning : Learns from labelled data
- Widely used classification technique in Machine Learning
- Finds the maximum margin hyperplane separating 2 groups of points
- We use R's implementation from library(e1071)

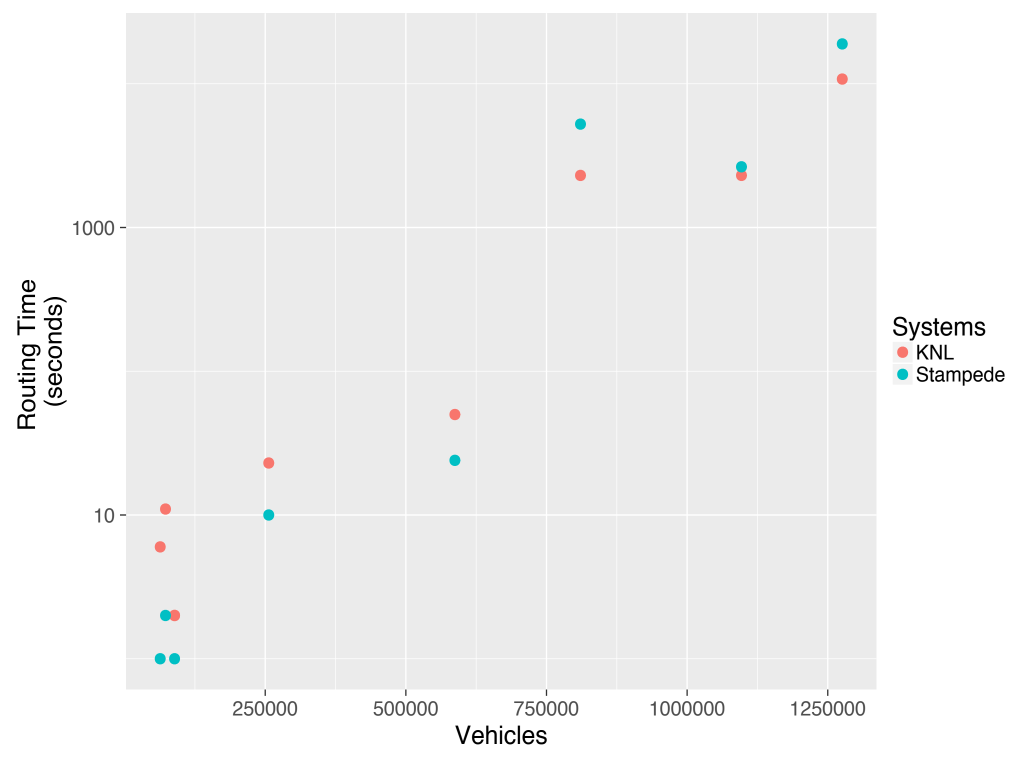
SVM

- Different Kernels employed to measure similarity between 2 samples
- We empirically explore
  - Linear
  - Polynomial
  - Radial basis
  - Sigmoid

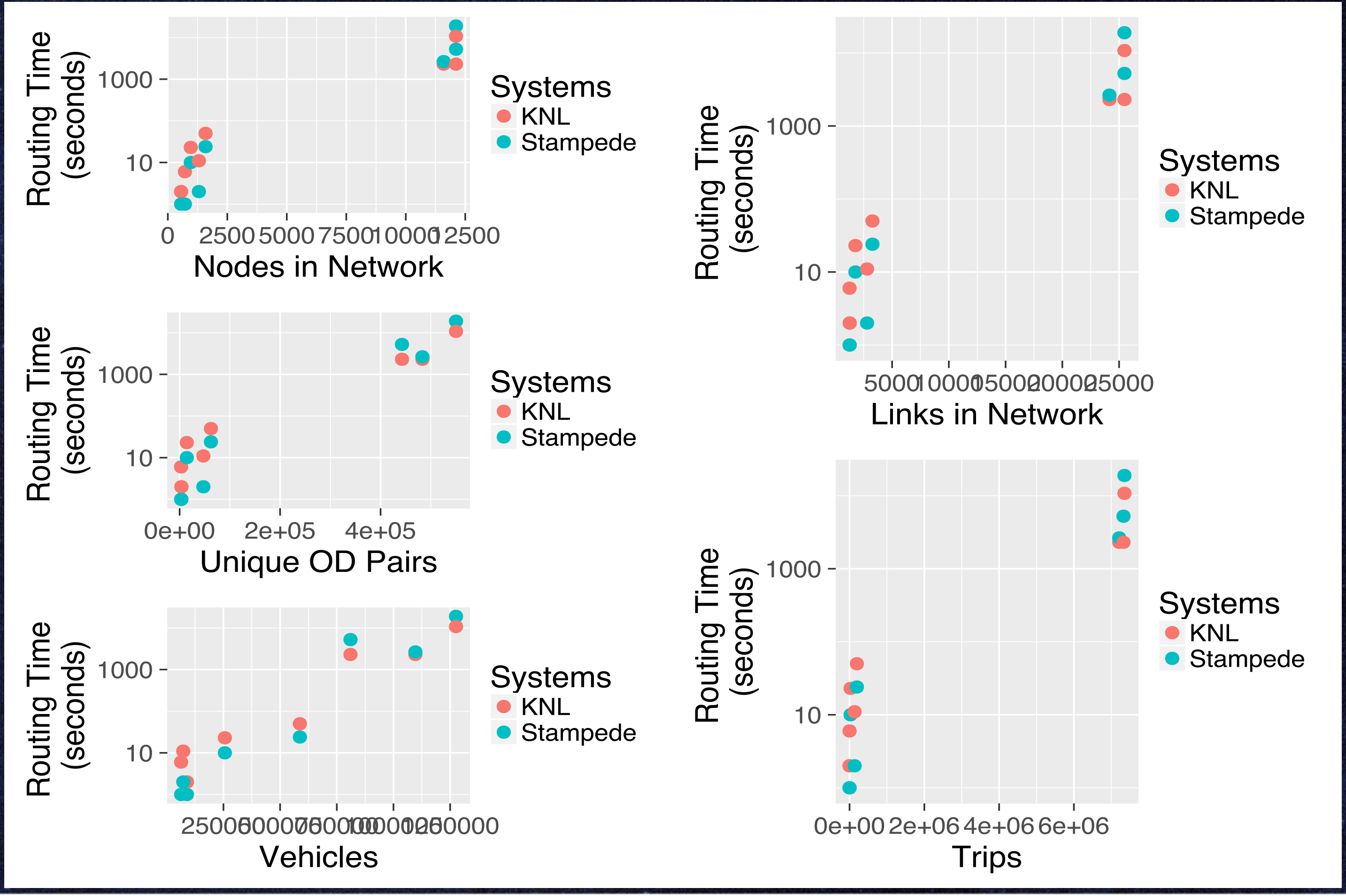
FEATURE SELECTION : DTA

- We select the following (7) problem features to the SVM model
- Hardware : Processing Power
  - (Total Number of Hardware Threads across all nodes) X (Speed of one core)
- Hardware : Memory
  - (Total memory across all nodes)
- Problem Size : Graph Topology
  - Nodes
  - Links
- Problem Size : Computation Size
  - Number of Unique Origin-Destination pairs
- Problem Size : Simulation Dynamics
  - Number of Vehicles in simulation
- Problem Size : Simulation Dynamics
  - Number of Trips in simulation

DTA SCALING



- Shown here is for illustration Scaling with the Vehicles feature
- Tested across 8 real world transportation networks
  - (*Included in the paper*)
- Neither system uniformly better for all data sets


Routing Time (seconds)
1000
10
0
2500
5000
Nodes in Network
Systems
KNL
Stampede
Routing Time (seconds)
1000
10
0e+00
2e+05
4e+05
Unique OD Pairs
Systems
KNL
Stampede
Routing Time (seconds)
1000
10
Vehicles
Systems
KNL
Stampede
Routing Time (seconds)
1000
10
Links in Network
Systems
KNL
Stampede
Routing Time (seconds)
1000
10
0e+00
2e+06
4e+06
6e+06
Trips
Systems
KNL
Stampede

DTA : SVM PREDICTION RESULTS

SVM Kernel	70% (Training) – 30% (Testing)	80% (Training) – 20% (Testing)
Linear	93.56	93.01
Polynomial	95.19	92.74
Radial Basis	95.49	96.17
Sigmoid	94.90	95.39

- Labelled dataset 21 points
- 5 labelled for KNL
- 16 labelled for Stampede
- 1000 fold cross-validation

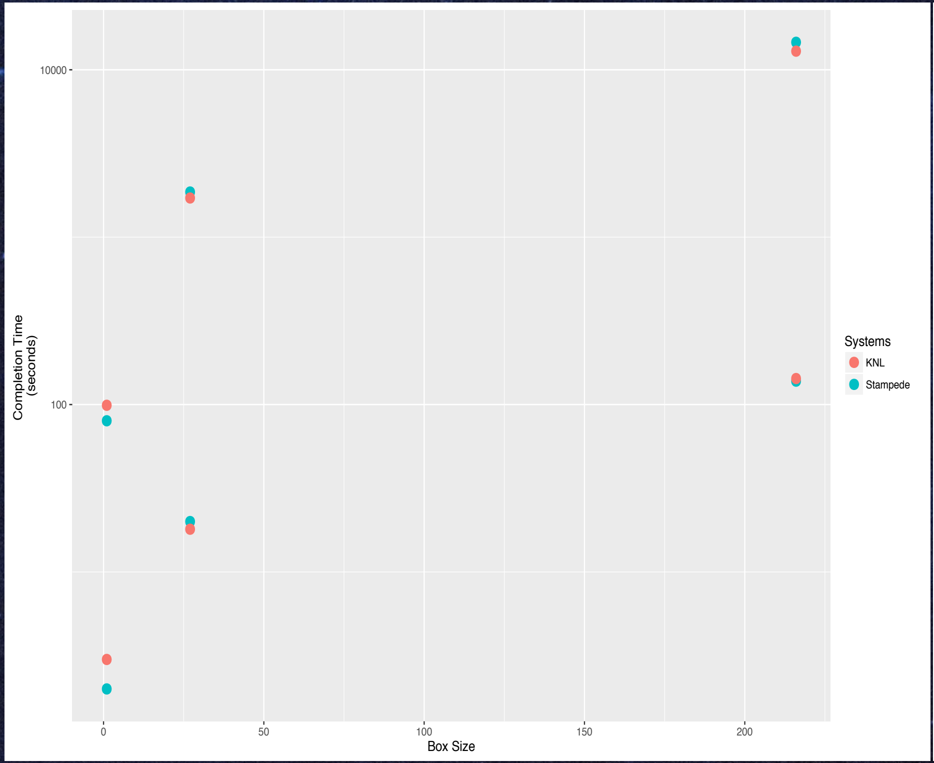
LAMMPS FEATURES

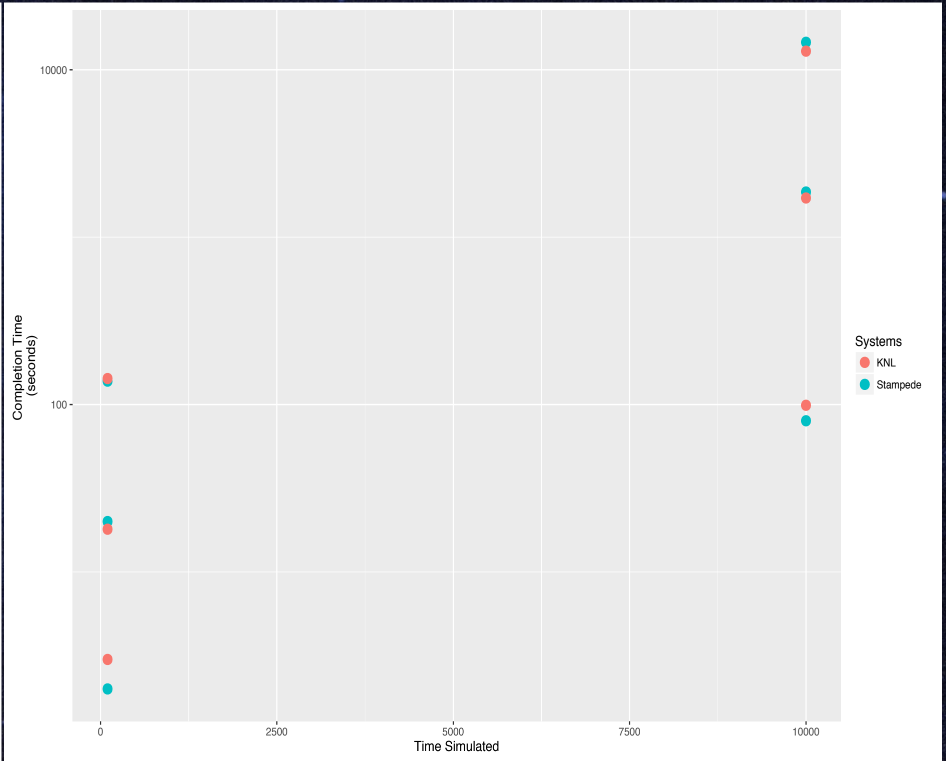
- We select the following (4) problem features to the SVM model

- Hardware : Processing Power
  - (Total Number of Hardware Threads across all nodes) X (Speed of one core)
- Hardware : Memory
  - (Total memory across all nodes)

- Problem Size : Box Size
- Problem Size : Time Simulated

LAMMPS SCALING





LAMPS : SVM PREDICTION RESULTS

SVM Kernel	70% (Training) – 30% (Testing)	80% (Training) – 20% (Testing)
Linear	71.01	73.975
Polynomial	71.68	78.95
Radial Basis	76.35	80.525
Sigmoid	73.25	77.6

- Labelled dataset 24 points
- 15 labelled for KNL
- 9 labelled for Stampede
- 1000 fold cross-validation

CONCLUSION

- Two very different application workloads
  - VISTA (Transportation Simulation, Graph Processing)
  - LAMMPS (Molecular Dynamics Simulation)
- Observed accuracy is promising
  - Datasets presented are small (20-30 samples in labelled data)
- Model would be more accurate with larger historical data
- Can optimize both System Utilization and End User Analysis time

FUTURE WORKS

- Test methodology on larger datasets
- Test methodology across different application/workload types
  - Hadoop/Spark based workloads
- Include more hardware features to do better prediction
  - Network Bandwidth
  - L2/L3 caches
  - Memory Bandwidth
- Use this within an application specific portal

THANKS

- Computations run on the NSF funded STAMPEDE cluster at
  - **Texas Advanced Computing Center**
- Partially funded by **XSEDE**
- Simulation framework and Datasets provided by
  - **Center for Transportation Research, UT Austin**

- Questions?