Computational Design of Polymers and Macromolecular Biomaterials

Arthi Jayaraman

Associate Professor

Dept. of Chemical & Biomolecular Engineering

and Dept. of Materials Science & Engineering

University of Delaware, Newark

HPC Symposium 2016

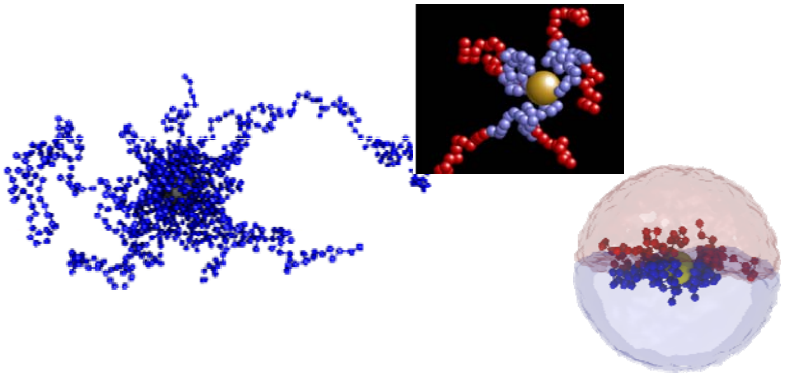
January 27th 2016

Research Group Website: http://udel.edu/~arthij/

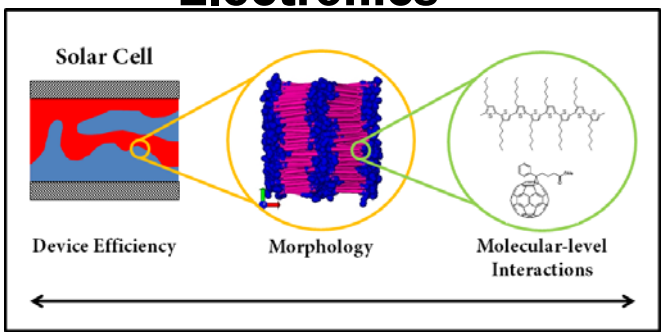
Polymer Modeling and Simulation in Jayaraman Group

How molecular features affect macroscopic morphology

Functionalized Nanoparticles in Polymer Nanocomposites

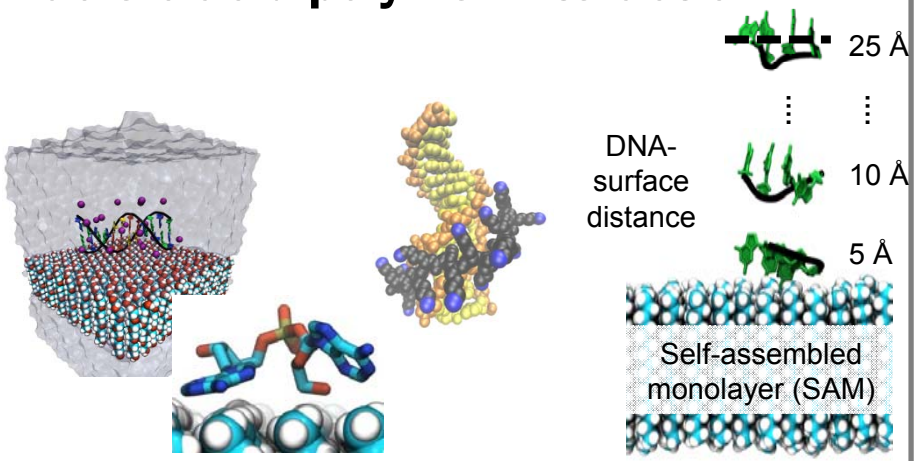


Conjugated Polymers for Organic Electronics

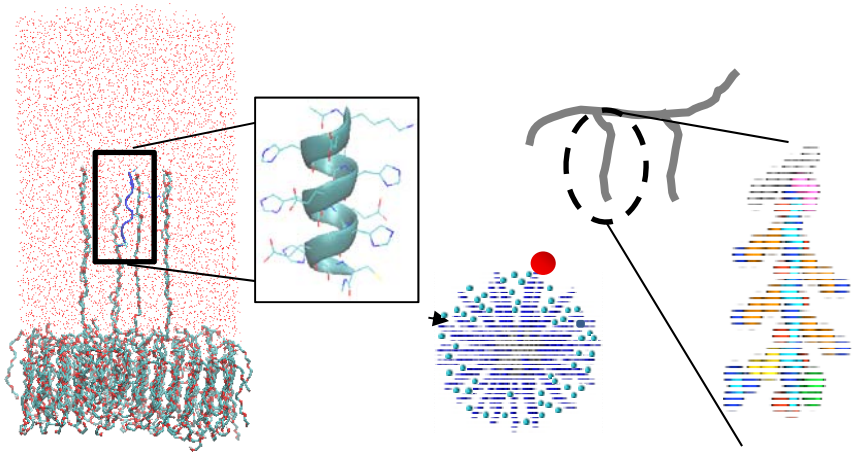


How molecular features impact macromolecular interactions

Nucleic acid-polymer interactions



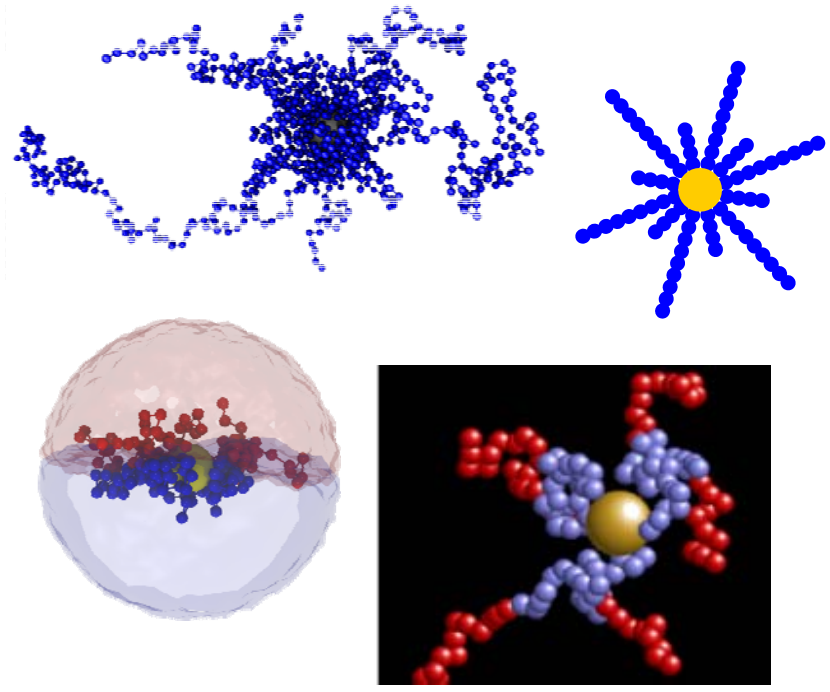
Peptide-polymer based biomaterials

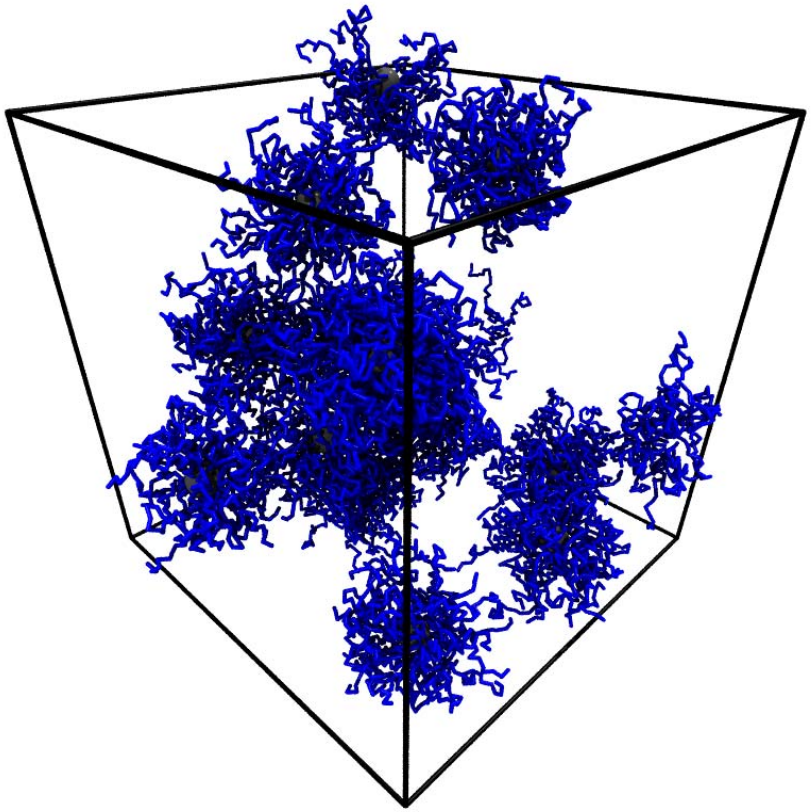


Polymer Nanocomposites:
Polymer grafted nanoparticles in a polymer matrix

Using theory and molecular simulations we are able to link features of polymer functionalization to particle dispersion/aggregation in a polymer matrix

Polymer Grafted Nanoparticles in a Polymer Matrix

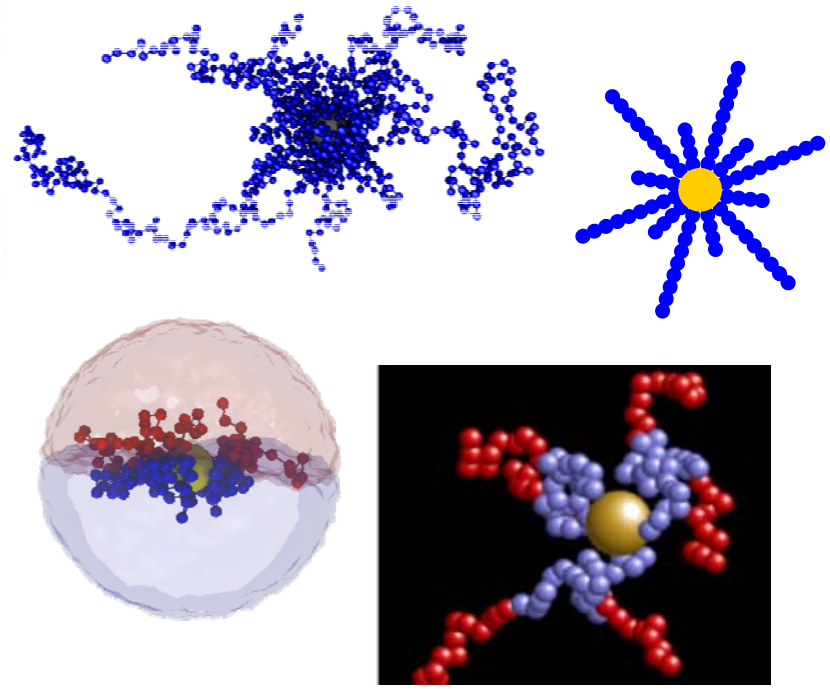


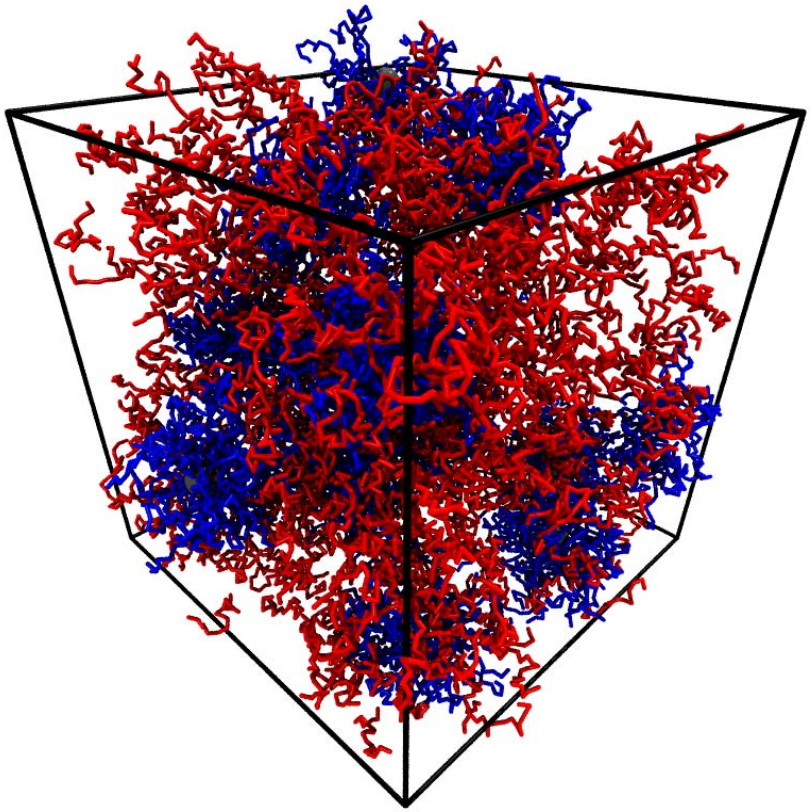


Polymer Nanocomposites:
Polymer grafted nanoparticles in a polymer matrix

Using theory and molecular simulations we are able to link features of polymer functionalization to particle dispersion/aggregation in a polymer matrix

Polymer Grafted Nanoparticles in a Polymer Matrix

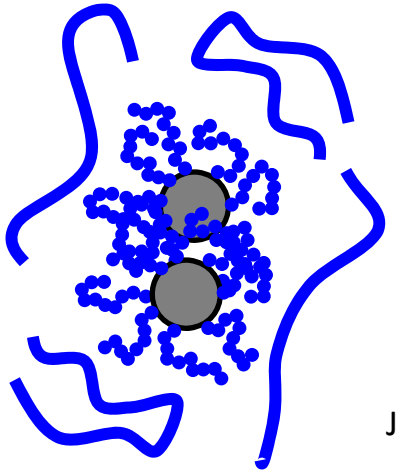


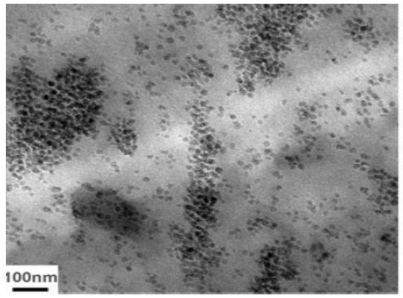


Physics Governing Morphology when Graft and Matrix Polymers are Chemically Similar

Particle *Aggregation*

Matrix MW > *Graft MW*

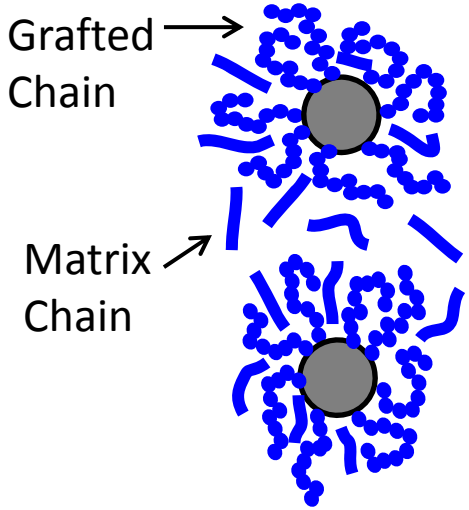


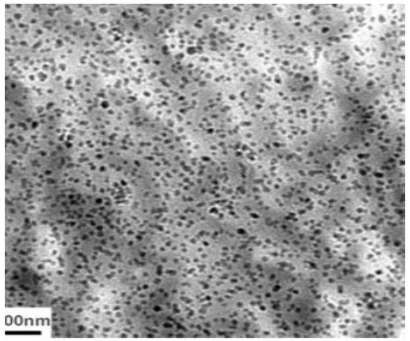


Bates and coworkers,
J. Polym. Sci. B: Polym. Phys.,
45, 2284 (2007)

Particle *Dispersion*

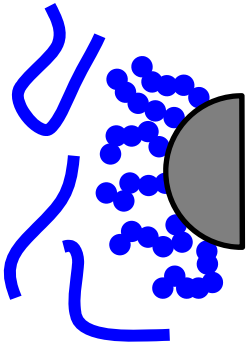
Matrix MW<< *Graft MW*



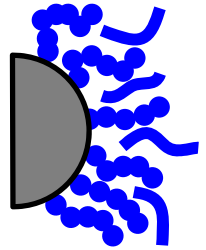


Bates and coworkers,
J. Polym. Sci. B: Polym. Phys.,
45, 2284 (2007)

Dry Brush

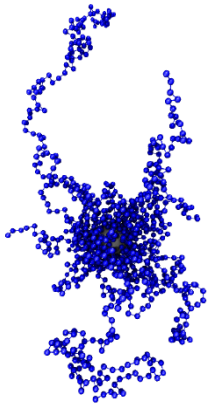


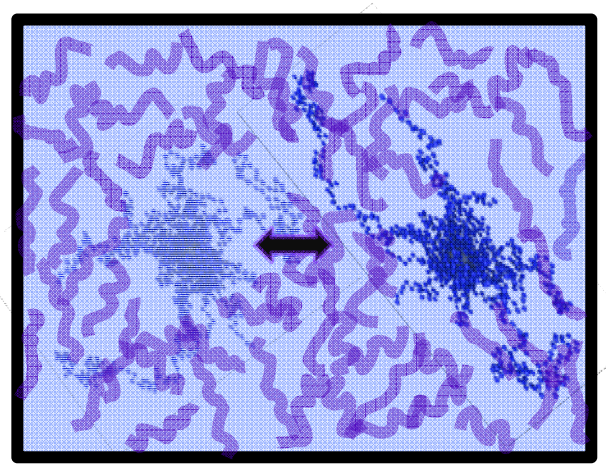
Wet Brush



Wetting/dewetting of the grafted layer tunes the polymer grafted particle **dispersion/aggregation** in the polymer matrix

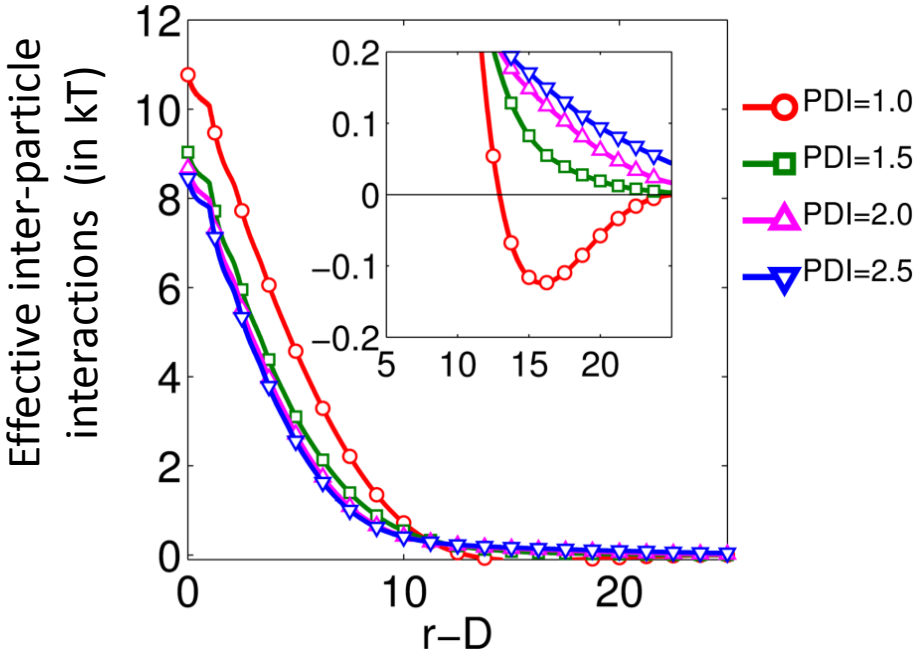
Designing polymer functionalization to improve nanoparticle dispersion in polymer matrix





Graft chemistry = Matrix chemistry

$N_{matrix} > N_{graft}$



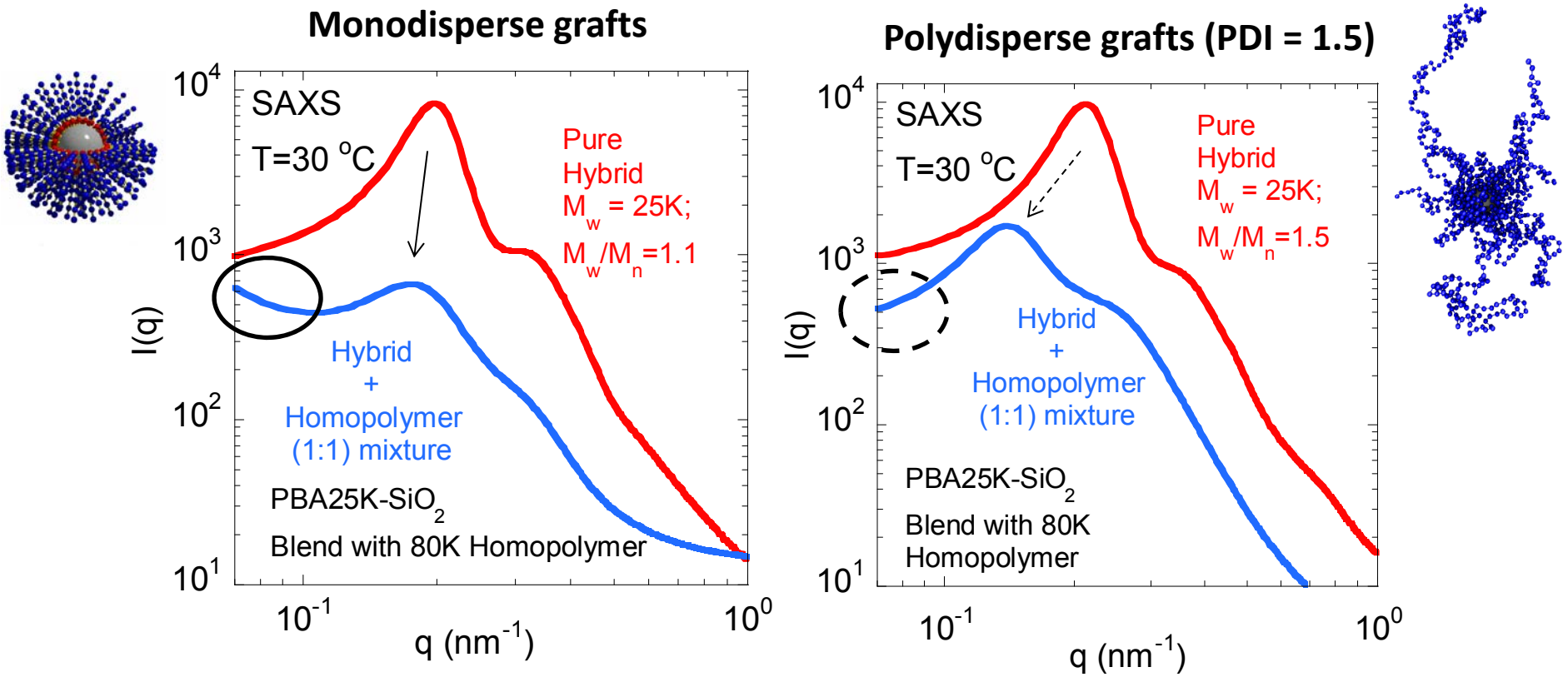
Polydispersity in graft polymer increases wetting of grafted layer by matrix chains, and thus improves particle dispersion in monodisperse polymer matrix.

Tyler Martin, P. Dodd, and A. Jayaraman Phys Rev Lett 110, 018301 (2013)

⇒ Our predictions were confirmed by experiments by Krishnamoorti and coworkers

Experiments showing agreement with our predictions

Results from laboratory of Prof. Ramanan Krishnamoorti at University of Houston



Polydisperse polymer grafted nanoparticles exhibit

a) ***Higher swelling (or wetting***) of grafted particles by matrix chains than monodisperse grafts

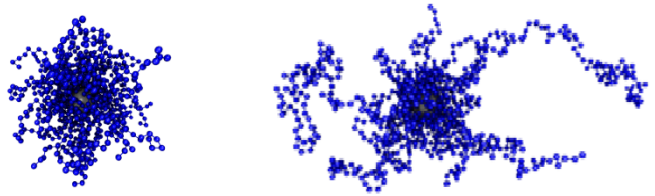
b) ***Reduced macrophase separation (or particle aggregation)***

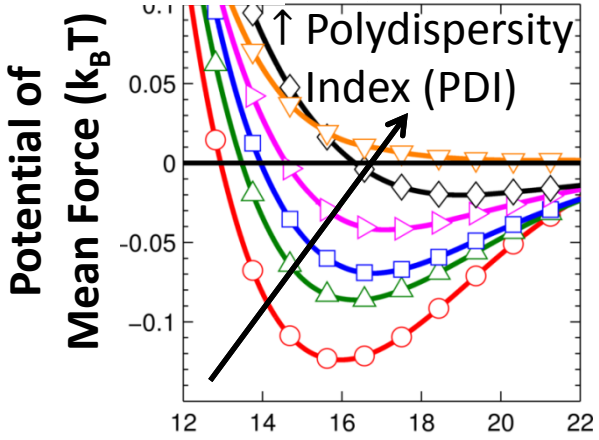
So far, experiments in agreement with our predictions.

Tailoring Entropic Driving Forces to Tune Morphology

Graft chemistry = Matrix chemistry

Monodisperse Polydisperse



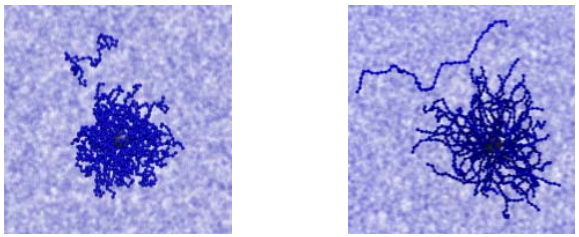


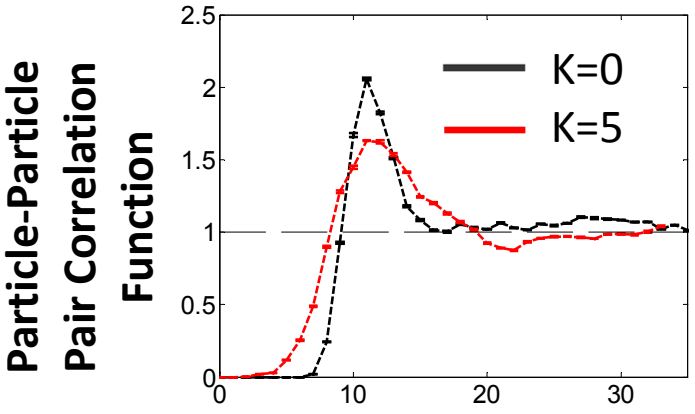
Interparticle Distance (d)

Increasing *graft length polydispersity* can *stabilize dispersions* of polymer grafted nanoparticles in a *monodisperse* polymer matrix.

Martin T.B., Dodd P.M., Jayaraman A., *Phys Rev Lett* 2013, 110 (1), 018301

Decreasing Polymer Flexibility



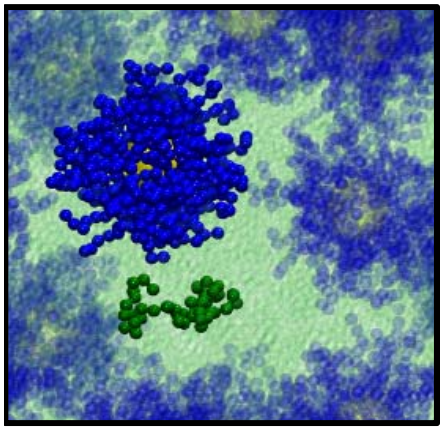


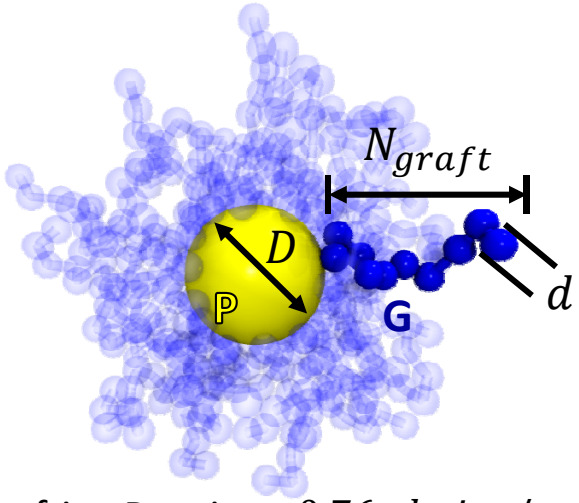
Interparticle Distance (d)

Decreasing *graft and matrix flexibility* can *stabilize dispersions* of polymer grafted nanoparticles in a *monodisperse* polymer matrix.

Lin B.†, Martin T.B.†, Jayaraman A., *ACS Macro Lett* 2014, 3 (7) 628-632

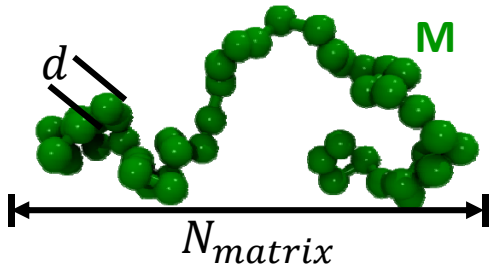
Chemically Dissimilar Graft and Matrix Homopolymers



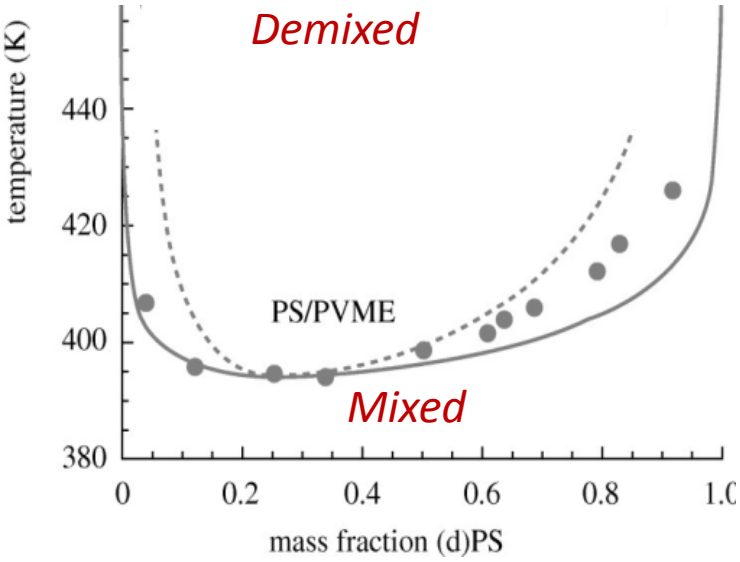


Grafting Density= $0.76\ chains/nm^2$

$N_{matrix} = 5N_{graft}$

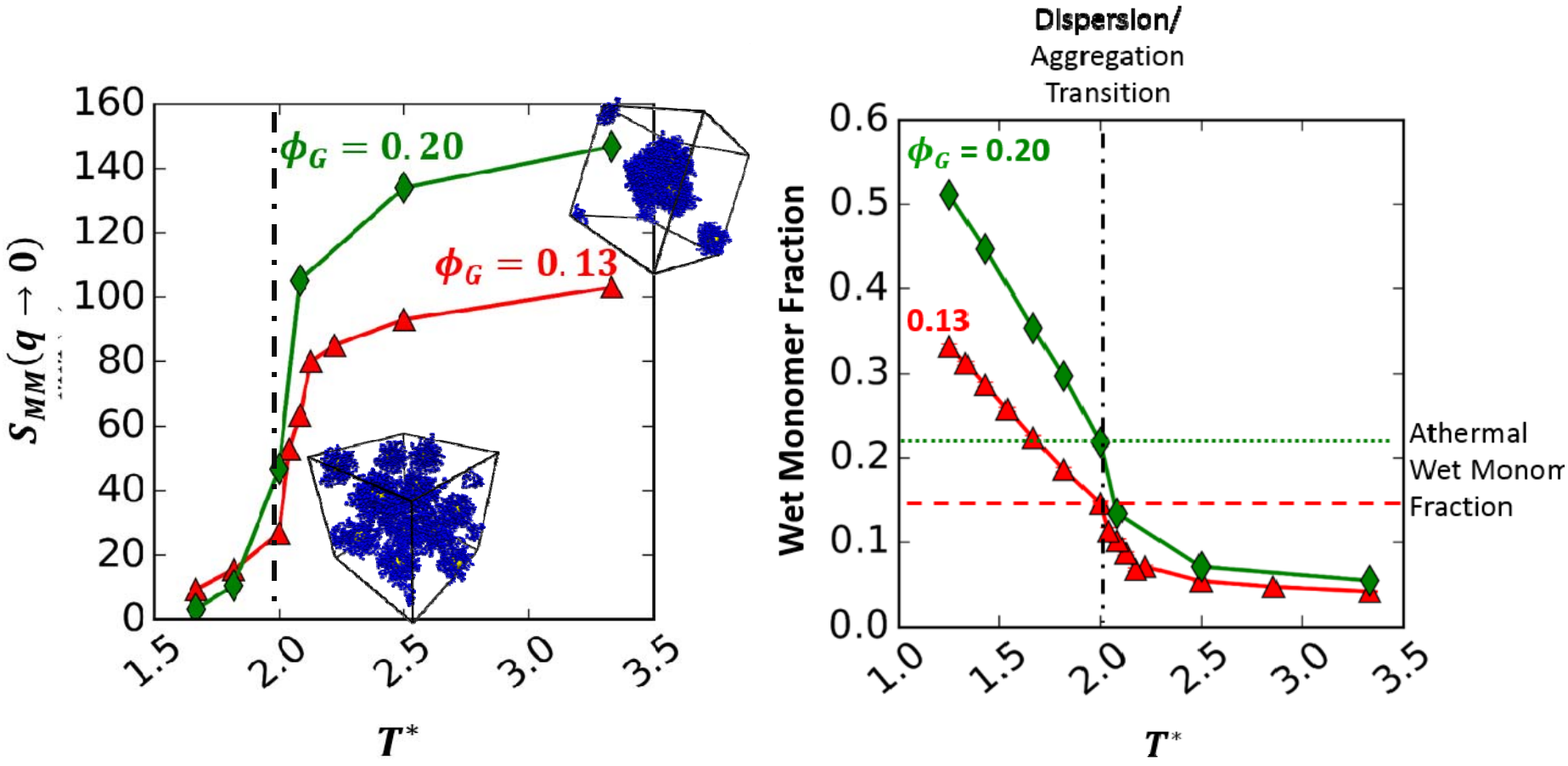


Example of LCST polymers



With attractive interactions between graft and matrix monomers, at low temperatures the enthalpically driven graft-matrix mixing should lead to particle dispersion even when $N_{matrix} >> N_{graft}$

Dispersion-Aggregation & Wetting Dewetting Transition



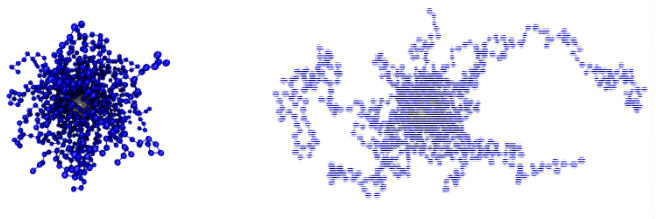
- Simulations and experiments show that wetting-dewetting transition is gradual and distinct from the sharp dispersion-aggregation.
- Critical wetting that marks onset of dispersion->aggregation is equal to the wetting found in corresponding athermal system.

Tyler B. Martin, K. Mongcopa, P. Butler, R. Krishnamoorti, A. Jayaraman *JACS* (2015) 137 (33), pp 10624–10631

Tailoring Thermodynamic Driving Forces to Tune Polymer Nanocomposite Morphology

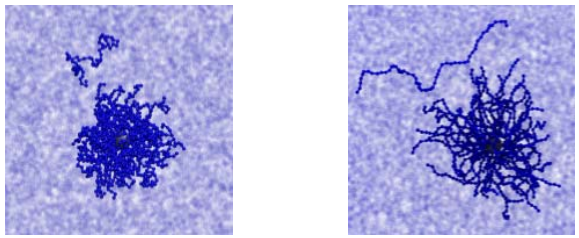
Tuning Entropic Driving Forces

Increasing Graft MW Dispersity



Martin T.B., Dodd P.M., Jayaraman A., *Phys Rev Lett* 2013, 110 (1), 018301

Decreasing Polymer Flexibility

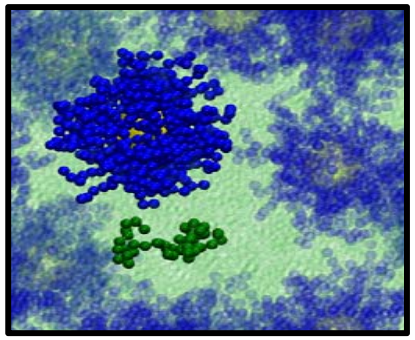


Lin B.†, Martin T.B.†, Jayaraman A., *ACS Macro Lett* 2014, 3 (7) 628-632

Tuning *mostly* Enthalpic Driving Forces

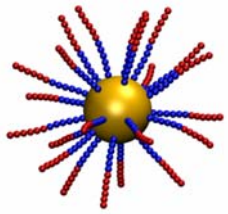
Chemically dissimilar graft and matrix polymers

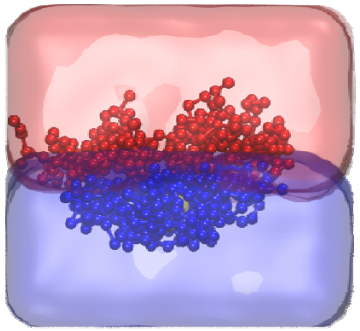
T. B. Martin, K. Mongcopa, P. Butler, R. Krishnamoorti, A. Jayaraman *JACS* (2015) 137 (33), pp 10624–10631



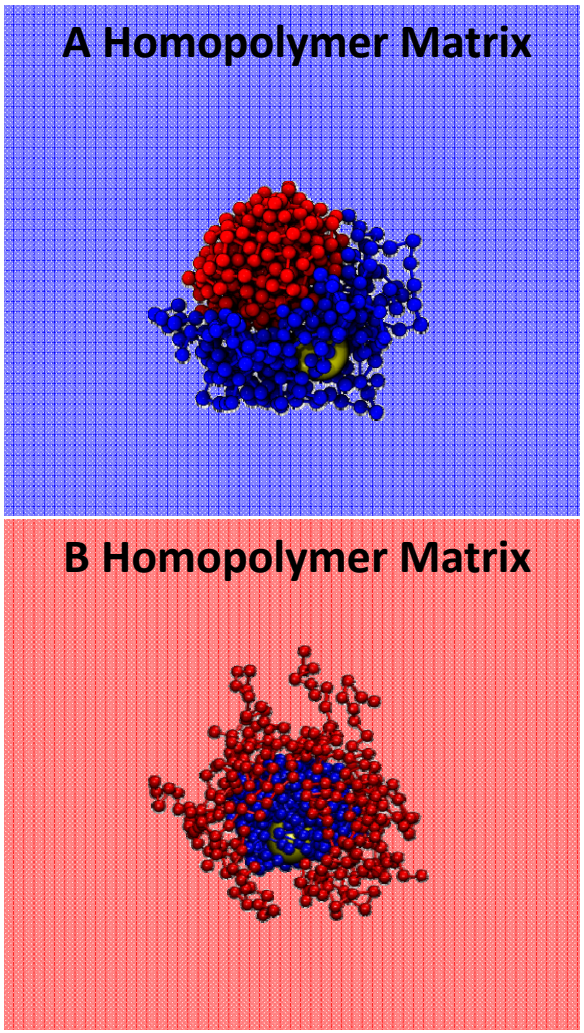
Copolymer grafted nanoparticles compatibilizing interfaces in homopolymer blends

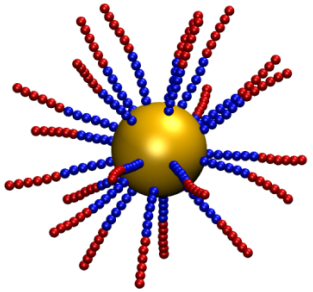
Estridge, C.E.; Jayaraman A. *ACS Macro Letters* 4(2) 155-159, 2015

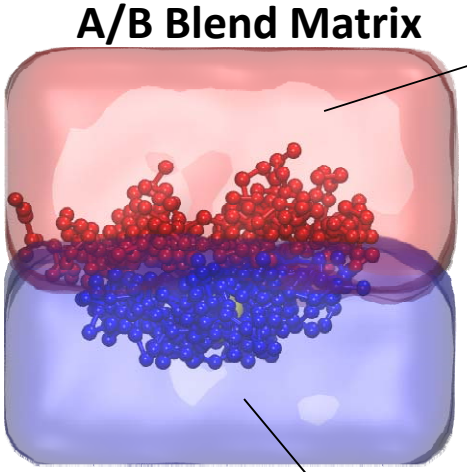


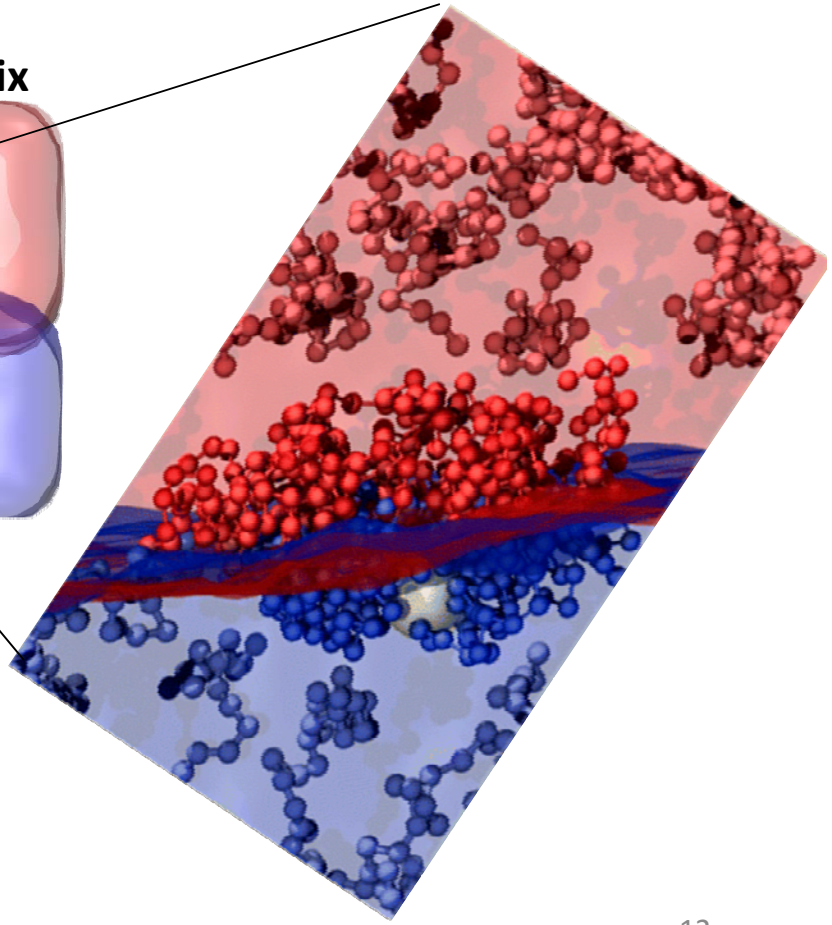


Diblock copolymer functionalization to stabilize interfaces







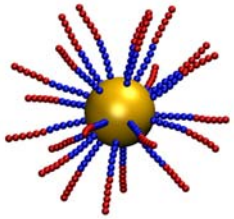


C. E. Estridge, A. Jayaraman ***ACS Macroletters*** (2015) 4 (2), 155

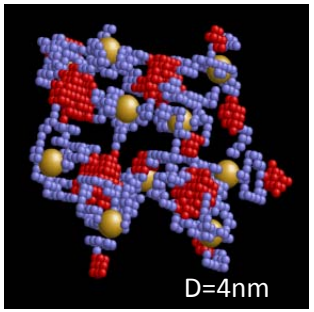
C. E. Estridge, A. Jayaraman, ***J Polymer Science B: Polymer Physics*** 2015,53,76-88

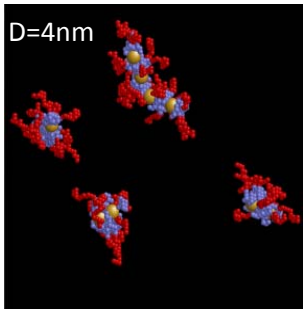
Designing polymer functionalization to direct nanoparticle assembly

Copolymer functionalization on the nanoparticles tuned to achieve desired **particle assembly and order**

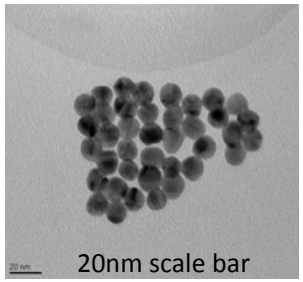


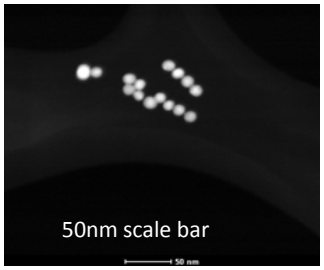
Particle Assembly



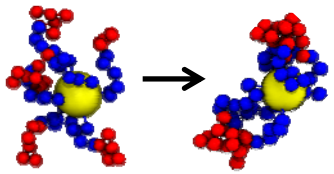


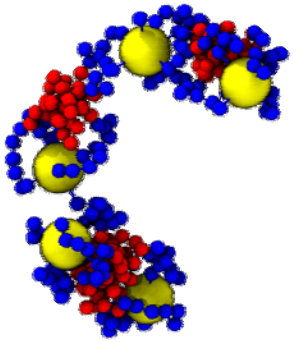
Experiments done in Dr. Park's lab at UColorado





Single Particle





Assembly of Multiple Particles

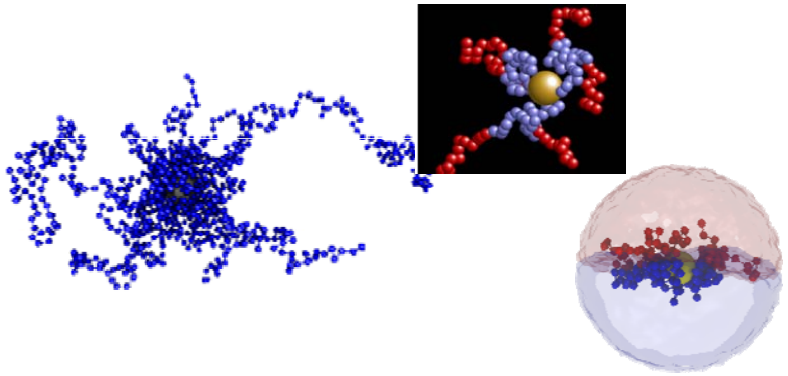
T. Martin, A. Seifpour and A. Jayaraman Soft Matter (2011) 7,5952-5964 (2011)

T. Martin, C. McKinney and A. Jayaraman Soft Matter (2013) 9, 155-169 2013

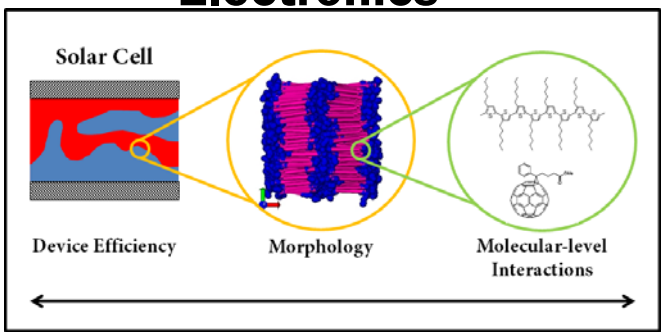
Polymer Modeling and Simulation in Jayaraman Group

How molecular features affect macroscopic morphology

Functionalized Nanoparticles in Polymer Nanocomposites

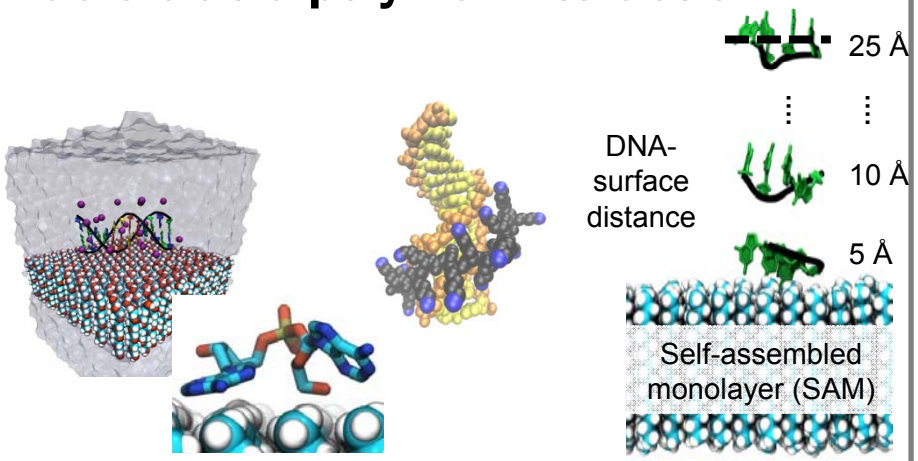


Conjugated Polymers for Organic Electronics

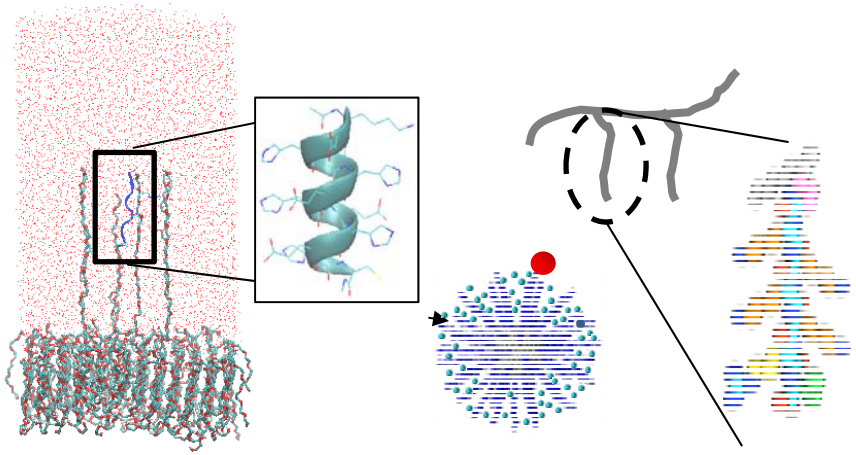


How molecular features impact macromolecular interactions

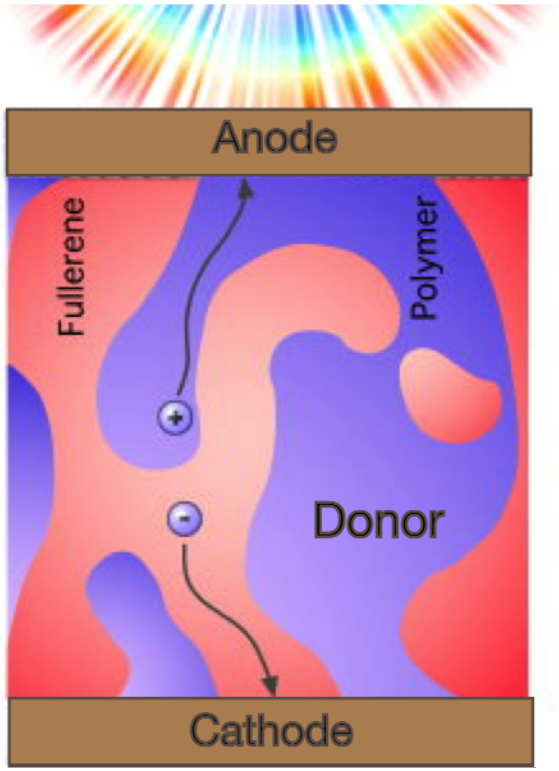
Nucleic acid-polymer interactions



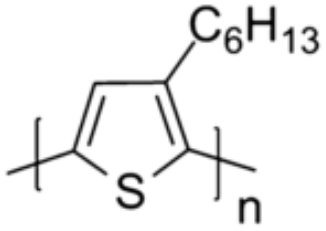
Peptide-polymer based biomaterials



Efficiency of organic photovoltaics depends upon active layer morphology

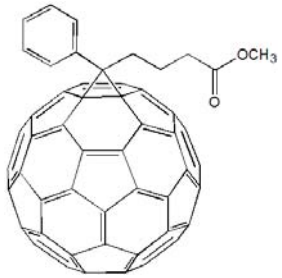


Donor: Conjugated polymer



e.g. P3HT

Acceptor: Fullerene derivative



e.g. PCBM

1) Excitons generated in donor material
2) Exciton dissociation at donor/acceptor interface
3) Charge transport to electrodes

Morphology (e.g. polymer crystallinity, domain connectivity, interfacial area) is important for device efficiency

http://blog.disorderedmatter.eu/2008/06/05/picture-story-how-do-organic-solar-cells-function

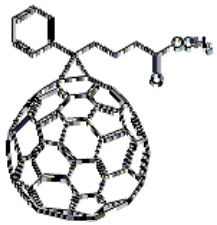
Challenge: Various Length Scales in Donor-Acceptor Morphology

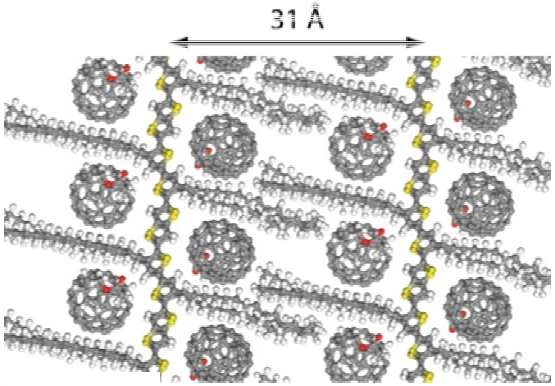
0.1nm- 5nm

>1nm < 100nm

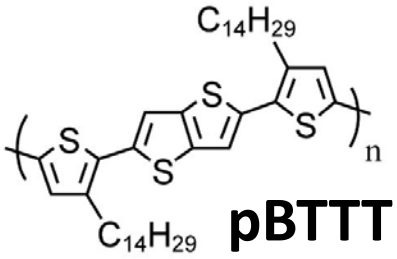
>100nm

PCBM

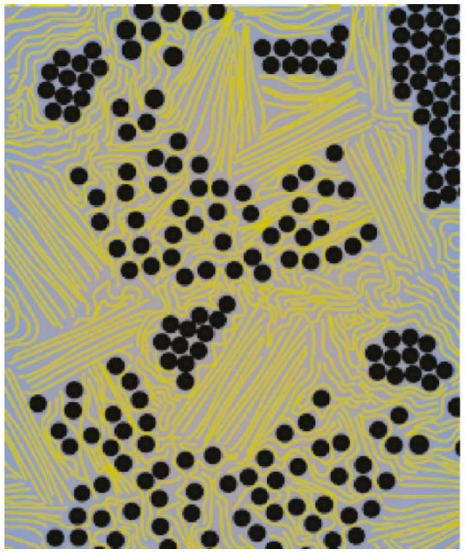




McGehee and coworkers, *Nano Letters*, 2009



pBTTT



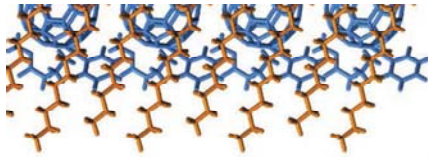
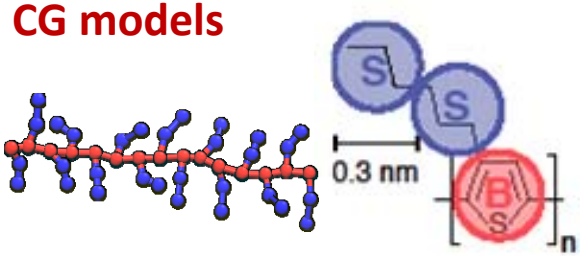
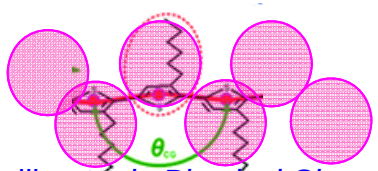
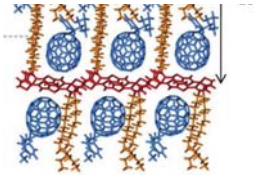
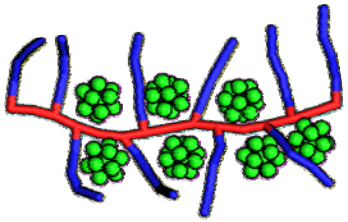
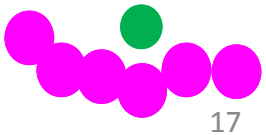
Yin and Dadmun, ACS Nano 2011

Active layer film in solar cell

Molecular-level detail, e.g. fullerene intercalation

Phase separation of donor and acceptor into domains (disorder→ order)

Comparison of Molecular Simulation Approaches

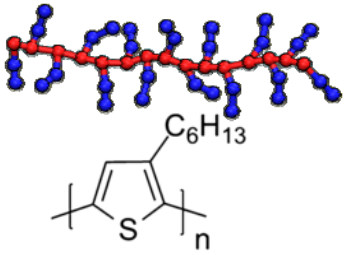
	Atomistic models	Intermediate resolution CG models	Highly Coarse-Grained models
	 McGehee and coworkers *Advanced Materials*, 2011		 Carrillo *et. al.*, *Physical Chemistry; Chemical Physics*, 2013
Length scale & Time scale	<10 nm << 100 ns	Up to 10-20nm 100s of ns	Up to 100nm > 1μs
Disorder to Order Transition	Computationally **not feasible** to capture Disorder to Order	Captures Disorder to Order	Captures Disorder to Order
Molecular packing, e.g. Crystallization, intercalation	Captures atomistic level packing *(initial configuration has to be ordered)* 	Captures backbone orientational order Captures intercalation 	Cannot capture most of the molecular level packing at the side chain- backbone level. 

Simulation Results with our Donor Polymer - Acceptor Model & Experimental Comparison

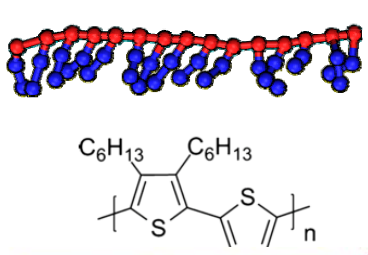
Simulation

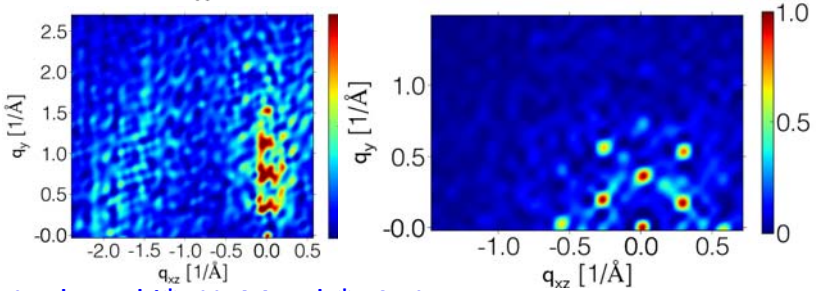
Agreement in Morphologies

P3HT



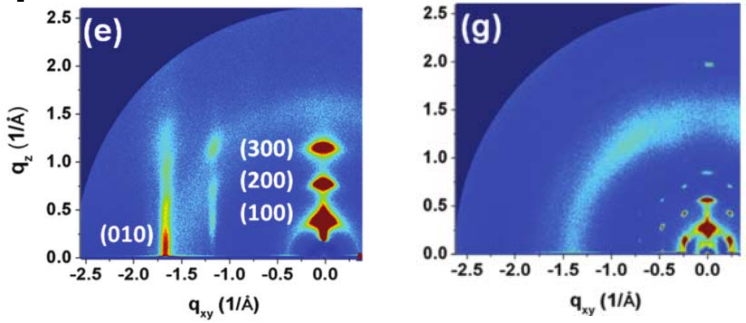
PDHBT





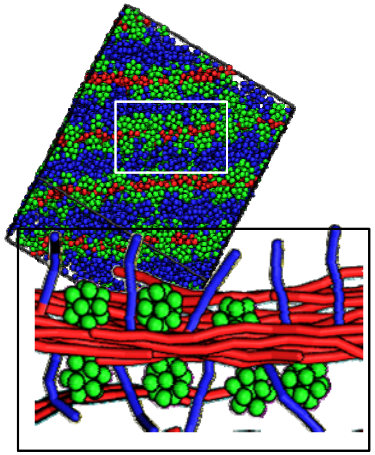
E. Jankowski*, H. Marsh*, A. Jayaraman, *Macromolecules* 46 (14) 5775-5785 (2013).

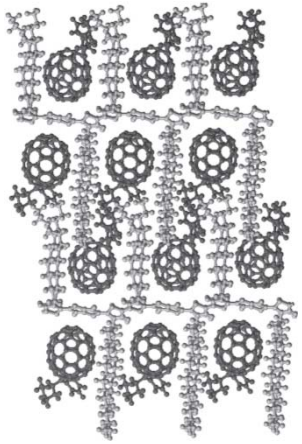
Experiment



Ko *et. al.*, *JACS.*, 2012

Observation of Intercalation

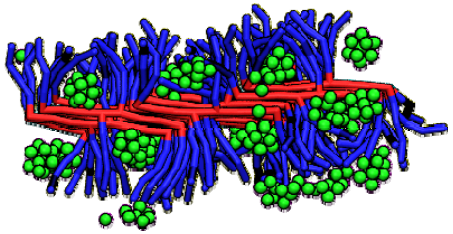


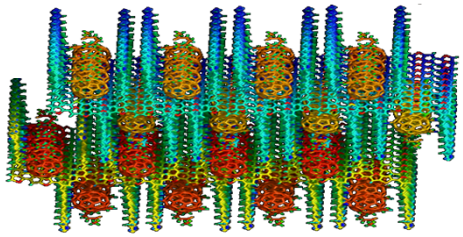


E. Jankowski*, H. Marsh*, A. Jayaraman, *Macromolecules* 46 (14) 5775-5785 (2013)

Miller et al. Adv. Energy Materials (2012)

Agreement in Morphologies





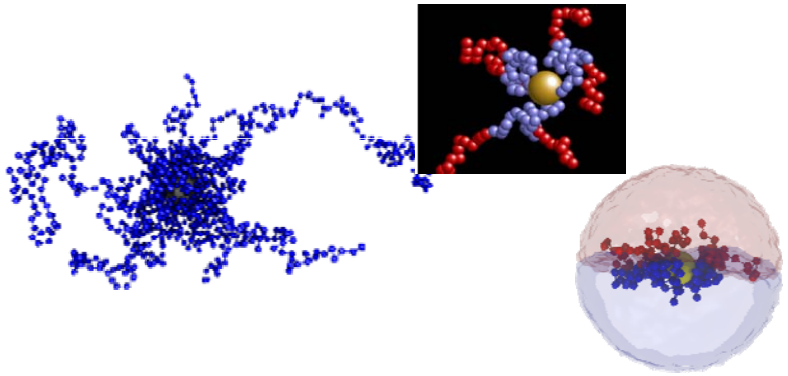
BTTT oligomers

Zhang, Briseno, Marsh, Jayaraman *JACS* 136 (52), 18120 (2014)

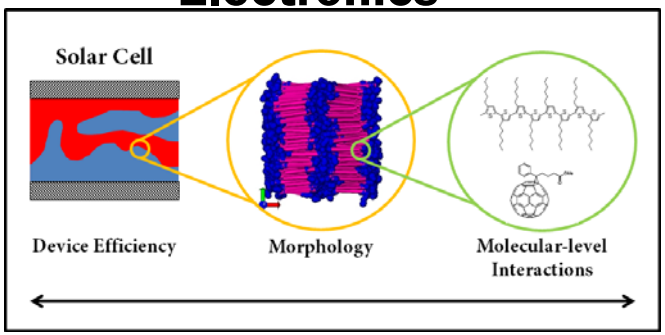
Polymer Modeling and Simulation in Jayaraman Group

How molecular features affect macroscopic morphology

Functionalized Nanoparticles in Polymer Nanocomposites

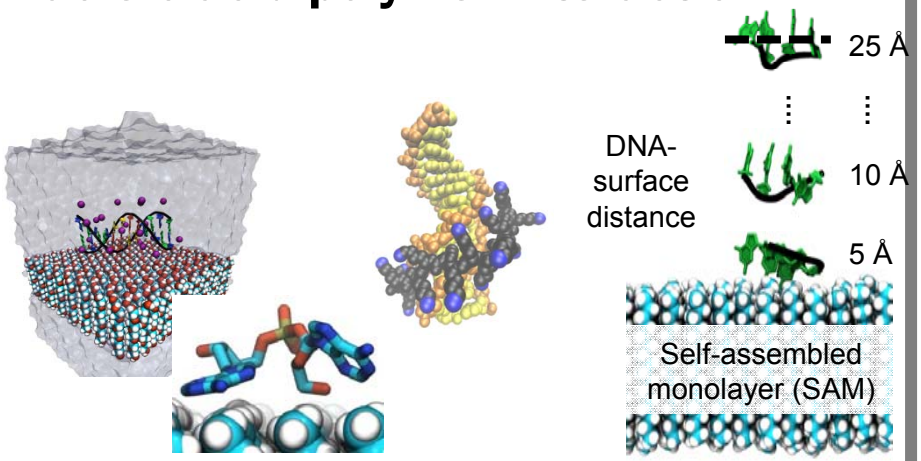


Conjugated Polymers for Organic Electronics

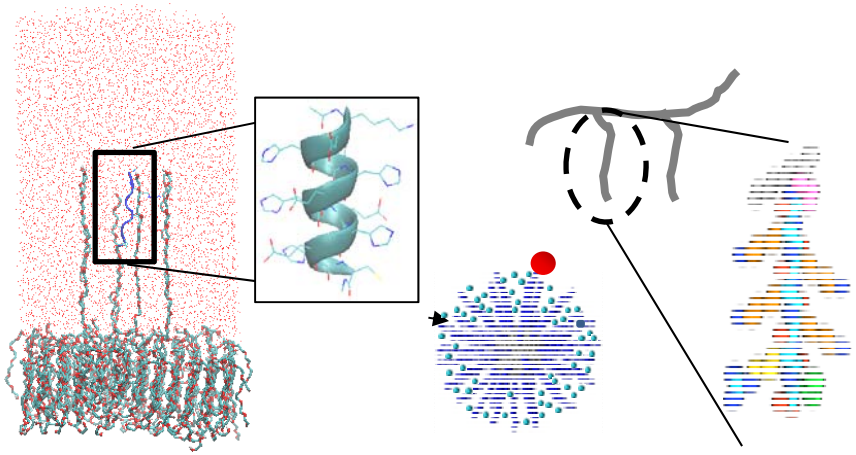


How molecular features impact macromolecular interactions

Nucleic acid-polymer interactions

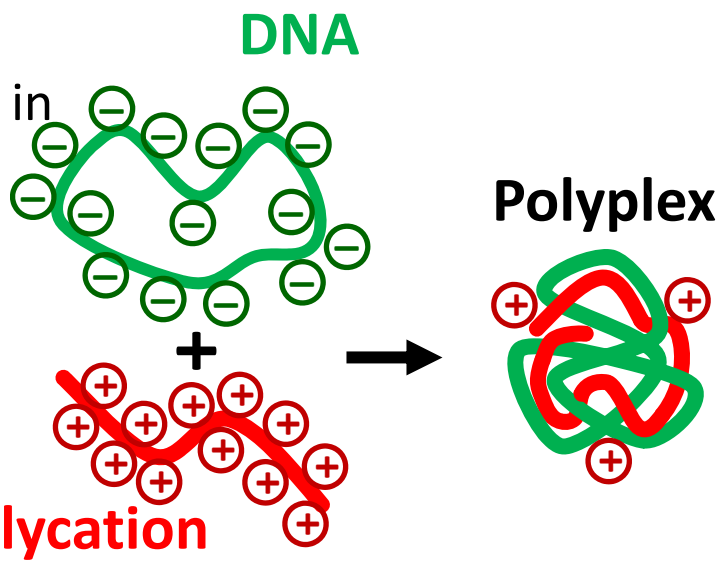


Peptide-polymer based biomaterials

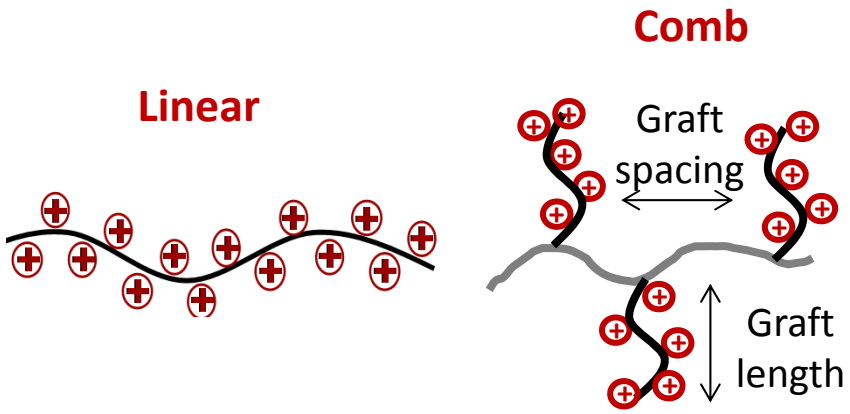


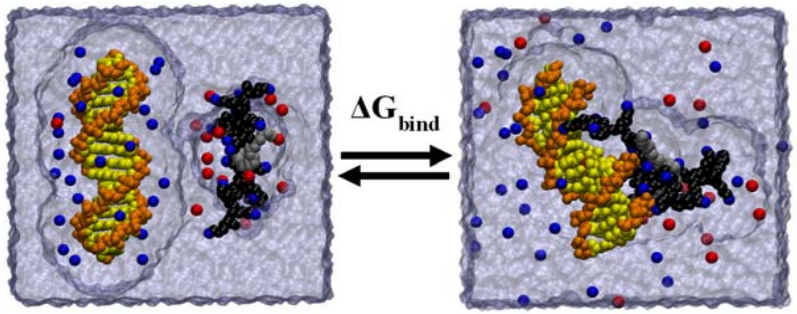
Polymers for Non-viral DNA Delivery

- Simulations linking cationic polymers architecture and chemistry to DNA-polymer complexation
- Polymer synthesis and in-vitro DNA transfection in T. Emrick's lab at UMass



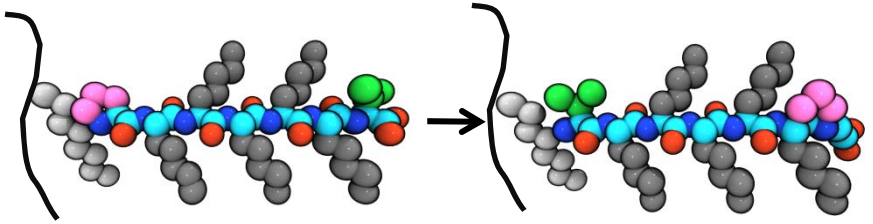
Effect of Varying Polymer Architecture On Polymer DNA Binding Thermodynamics





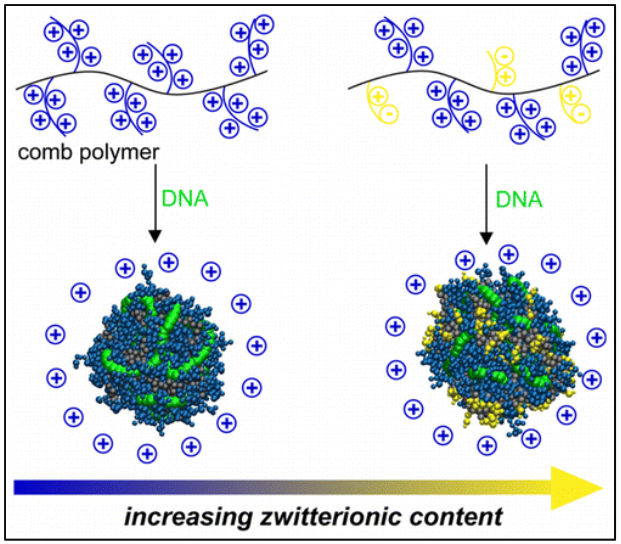
R. Elder, T. Emrick, and A. Jayaraman, *Biomacromolecules* 12 (11), 3870 (2011)

Effect of Varying Oligopeptide-based Comb-Polymer Chemistry

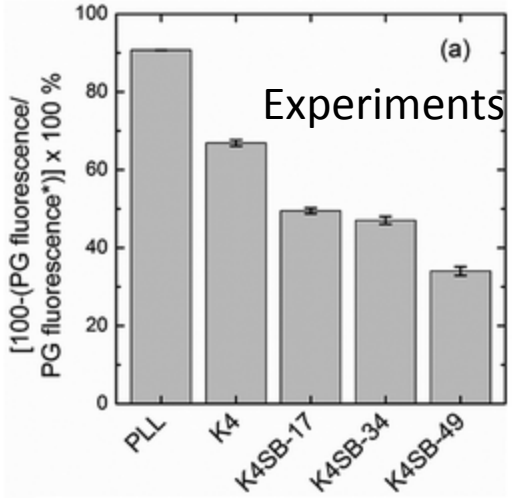


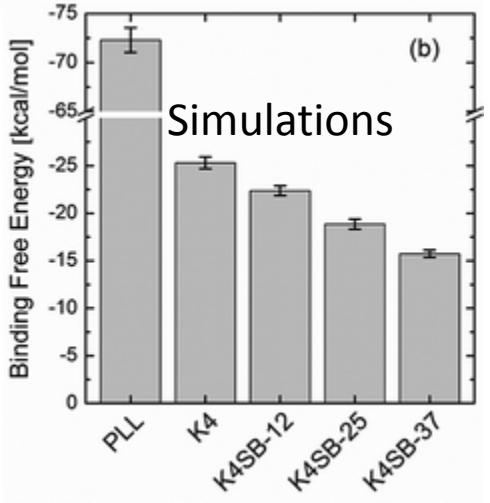
R. M. Elder and A. Jayaraman, *J. Phys. Chem. B.* (2013) 117 (40), pp 11988–11999

Zwitterion containing polymers for DNA delivery



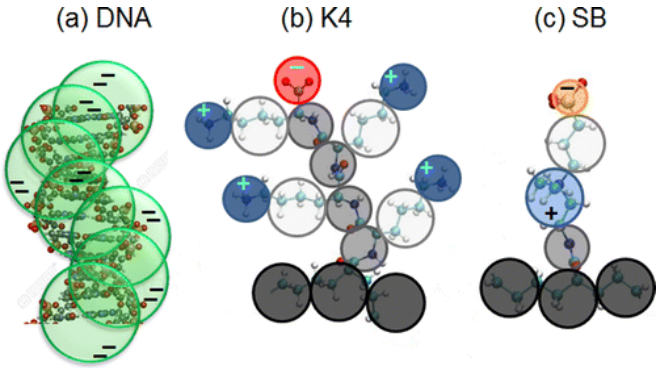
Then, established the effect of sulfobetaine content on polymer-DNA complexation- thermodynamics & structure

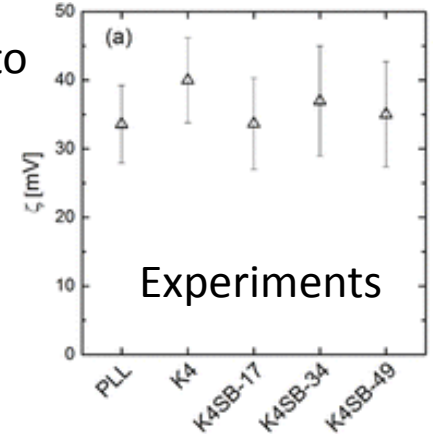


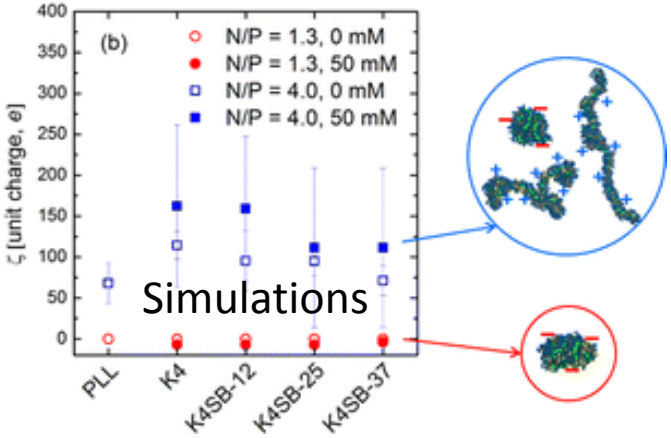


Ahmad Ghobadi, R. Letteri, T.Emrick, A.Jayaraman, *Biomacromolecules (2016)*
Article ASAP

We first developed appropriate models to simulate relevant length and time scales



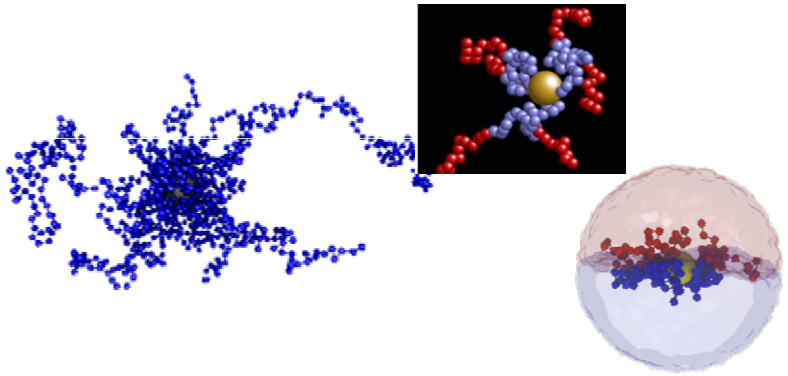




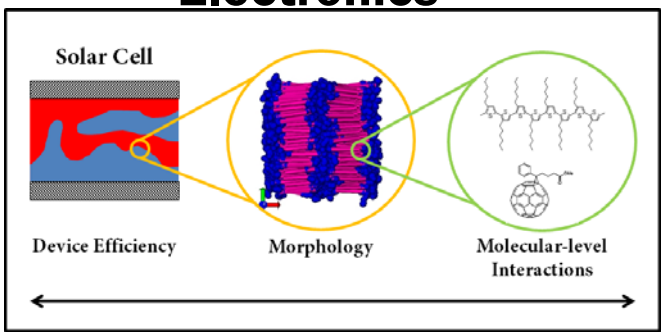
Polymer Modeling and Simulation in Jayaraman Group

How molecular features affect macroscopic morphology

Functionalized Nanoparticles in Polymer Nanocomposites

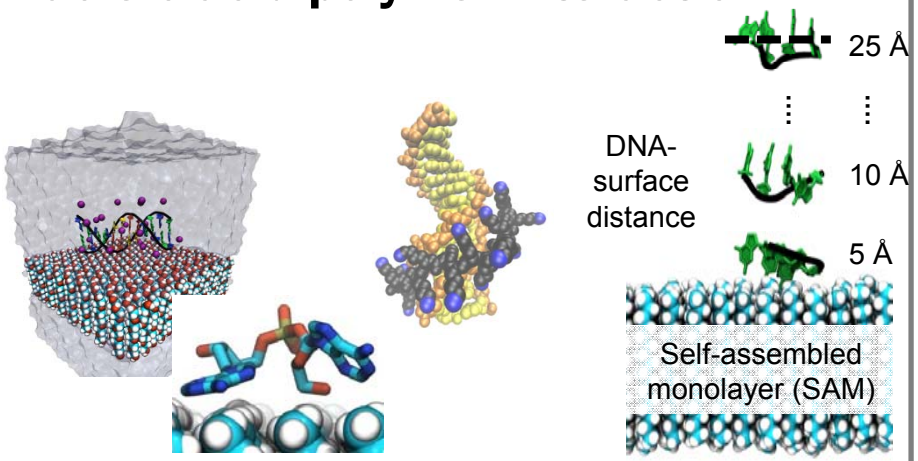


Conjugated Polymers for Organic Electronics

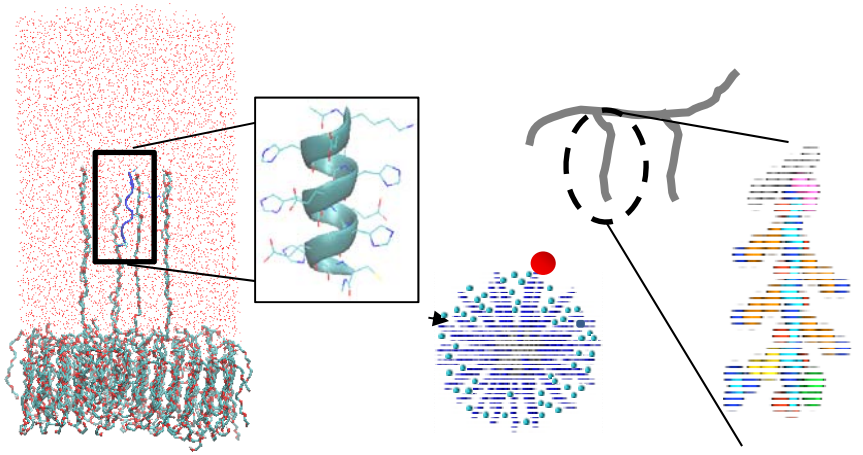


How molecular features impact macromolecular interactions

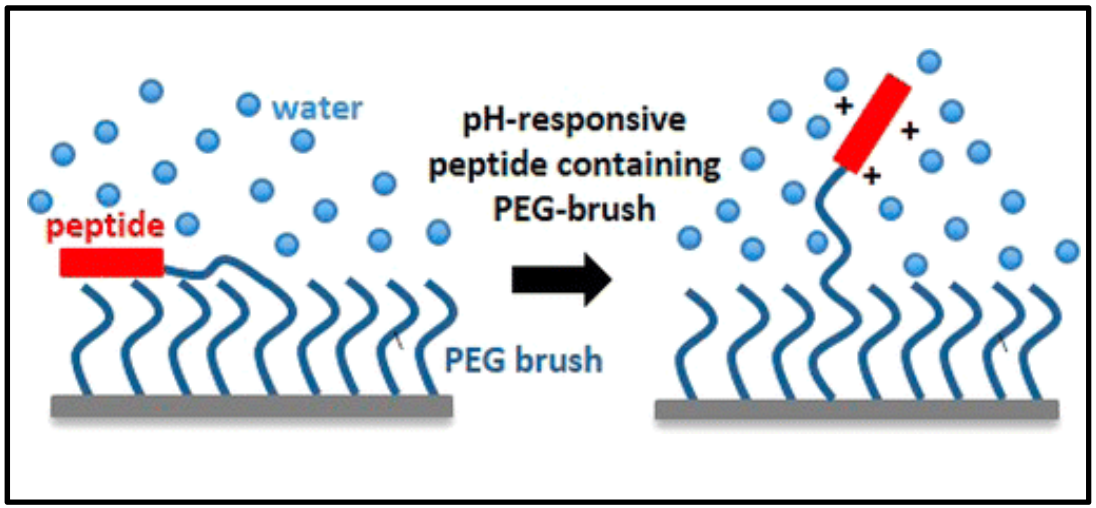
Nucleic acid-polymer interactions



Peptide-polymer based biomaterials



Peptide-Polymer Based Biomaterials

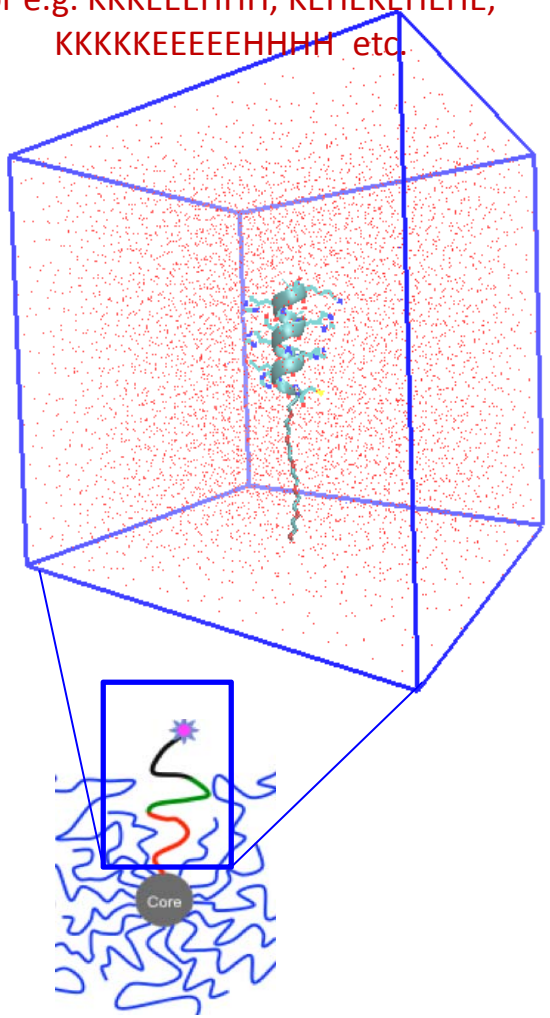


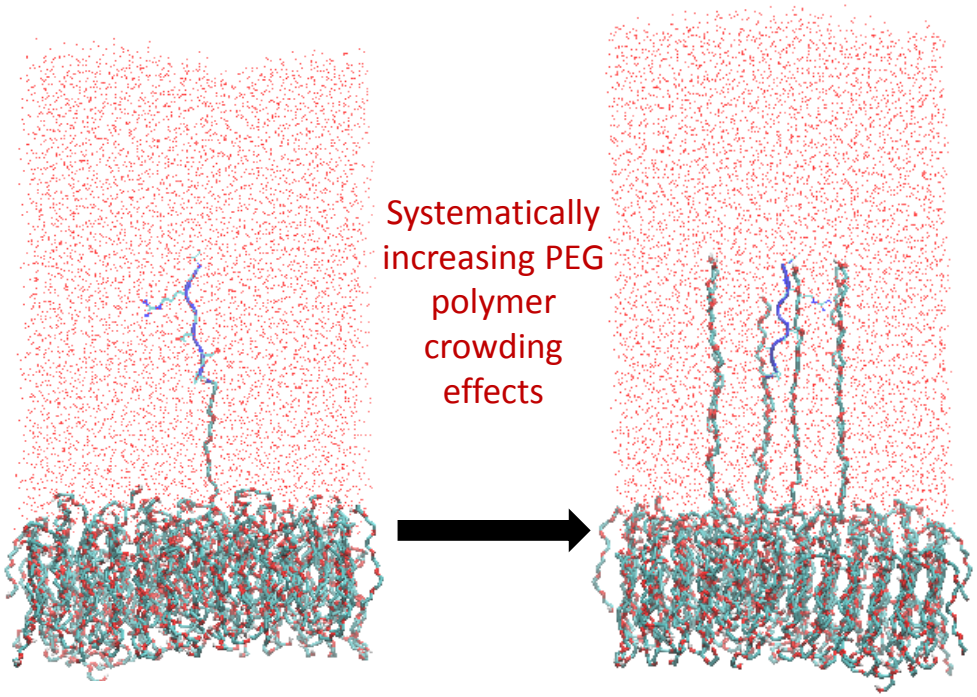
Computational Design of Oligopeptide Containing Poly(ethylene glycol) Brushes for Stimuli-Responsive Materials

Francesca Stanzione and Arthi Jayaraman
Journal of Physical Chemistry B *119* (42), 13309-13320 **2015**

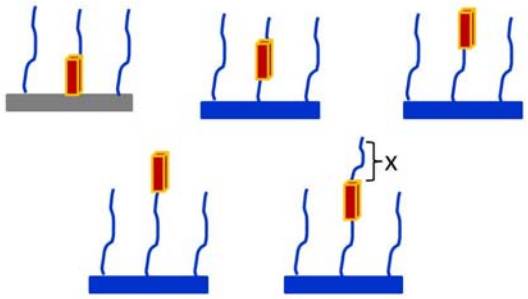
Computational Approach Scan of a Large Design Space

Systematically varying peptide length, sequence and composition
For e.g. KKKEEEHHH, KEHEKEHEHE, KKKKKEEEEEHHHH etc.





Systematically varying peptide placement within the PEG

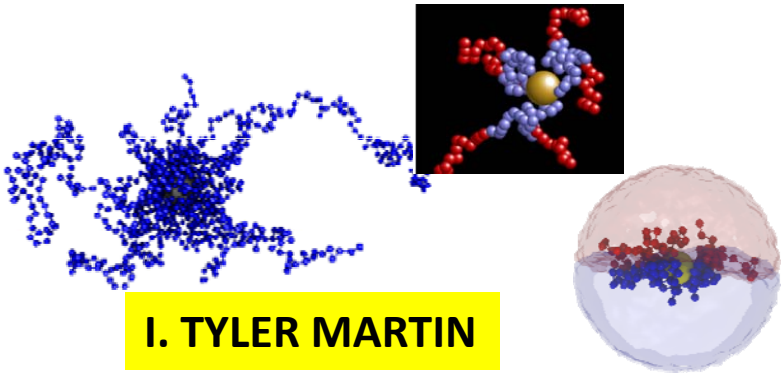


F Stanzione and A Jayaraman, *J. Phys Chem B 119* (42), 13309-13320 2015

Polymer Modeling and Simulation in Jayaraman Group

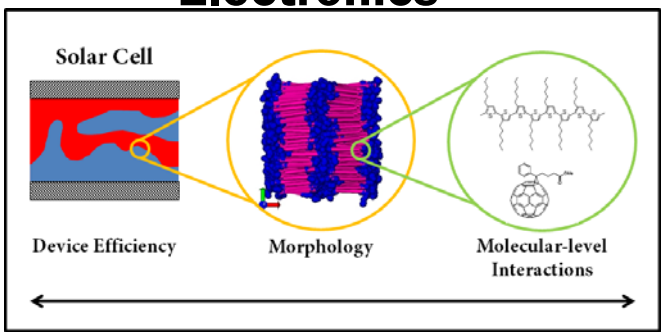
How molecular features affect macroscopic morphology

Functionalized Nanoparticles in Polymer Nanocomposites



I. TYLER MARTIN

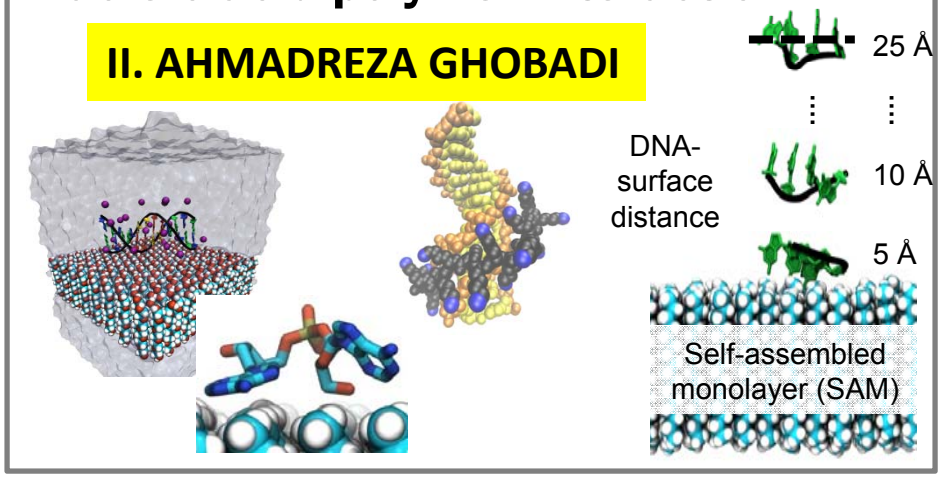
Conjugated Polymers for Organic Electronics



How molecular features impact macromolecular interactions

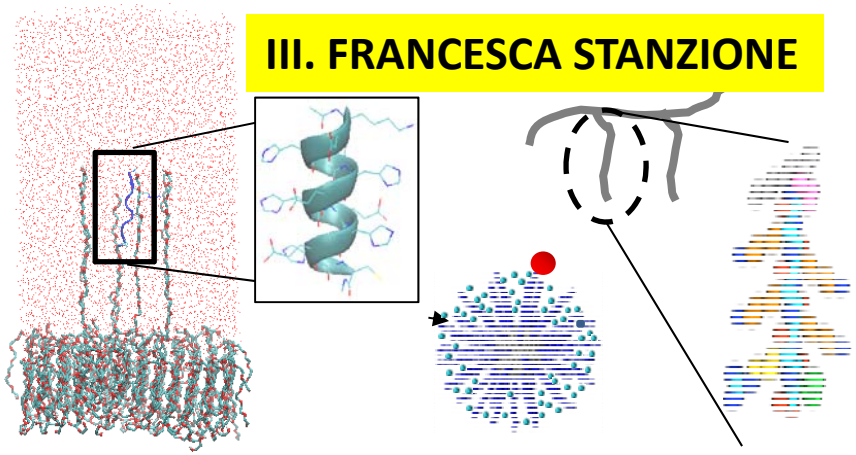
Nucleic acid-polymer interactions

II. AHMADREZA GHOBADI



Peptide-polymer based biomaterials

III. FRANCESCA STANZIONE



http://udel.edu/~arthij/

Jayaraman Research Group

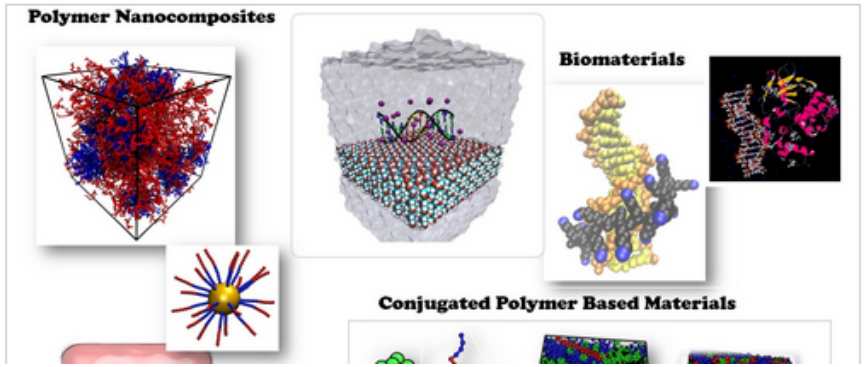
Theory and Simulations of Polymers and Biomaterials

Home Research Publications Members News Talks

Welcome!

The Jayaraman research group is lead by Prof. Arthi Jayaraman, Associate Professor, Chemical and Biomolecular Engineering and Materials Science and Engineering at University of Delaware. Our research interests lie in understanding molecular-level phenomena governing complex biological processes and material science problems using theoretical and simulation techniques.

Current Research Focus



Recent News

Our collaborative paper with Todd Emrick and coworkers (at UMass) on accepted in Biomacromolecules!

Tyler has been selected as a finalist for the Padden symposium at APS March meeting 2016!

Research highlight on our JACS paper!Link US Department of Energy also tweets about this work Link

Arthi has been selected as Princeton University's Saville Lecturer for 2015-16!

Current members of Jayaraman group

Tyler Martin - PhD student, Polymer Nanocomposites

Joshua Condon - PhD student, Peptide-Polymer Biomaterials

Thomas Gartner - PhD student, Polymer Nanocomposites

Dr. Francesca Stanzione- Peptide based biomaterials

Dr. Ahmad Ghobadi - Nucleic acid based biomaterials

Christopher Knieste - undergrad

Some of the group alumni and their current positions

- Arezou Seifpour – PhD 2013 – **Intel** (May 2013-)
- Robert Elder PhD Dec 2013 – **Army Research Lab** (Jan 2014 -)
- Carla Esteridge PhD May 2015 – **Boeing Research** (July 2015-)
- Hilary Marsh PhD May 2015 – NREL
- Charles Starbird MS 2012 – **Eastman** (June 2012-)
- Eric Jankowski Postdoc 2012-13 – Asst. Prof. at Boise State (2015-)
- Nitish Nair Postdoc 2009-11 – **Shell** (2011-)
- D. Zhang Postdoc 2011-12 – **UT Dallas** (2012-)

Overview of Using GPUs for Molecular Dynamics Simulations

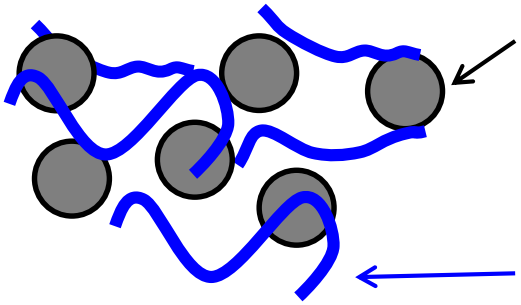
Tyler B. Martin

Advisor: Prof. Arthi Jayaraman

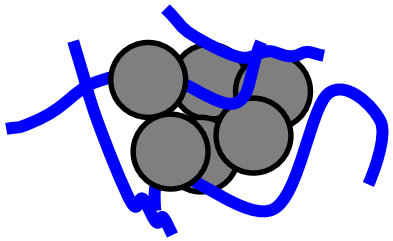
January 27, 2016

Polymer Nanocomposite Morphology

Dispersed Morphology



Aggregated Morphology



Polymer nanocomposite morphology dictates macroscopic properties

Mechanical Reinforcement

sae.org

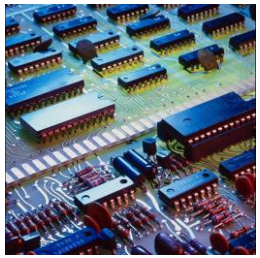
Contamination Barriers

placon.com

Aircraft Coatings

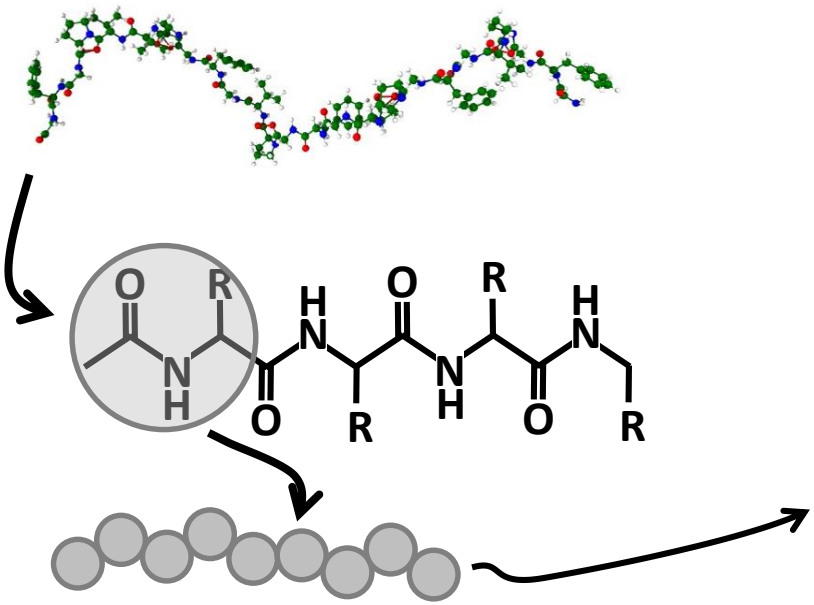
wikimedia.org

Microelectronics

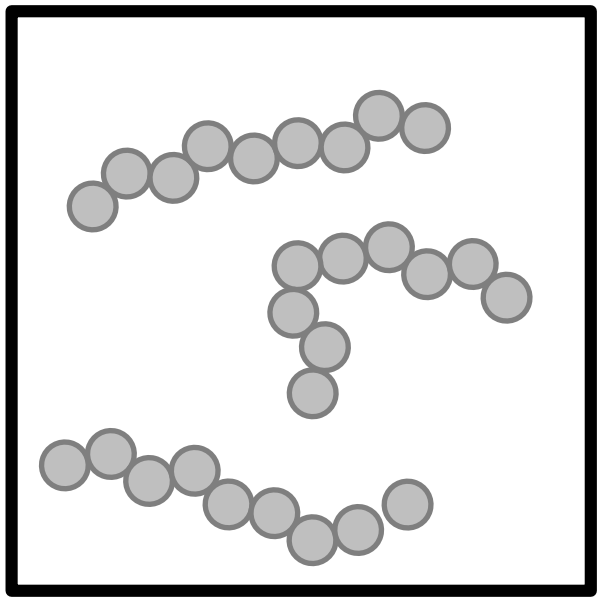


Chlorineremoval.com

How do we simulate polymer nanocomposites?



Molecular Dynamics Simulation



The force calculation in MD scales $O(N^2)$ where N is the number of atoms.

This is the dominant bottleneck in these simulations

MD Simulation Steps

For each bead in simulation…

1. Calculate force on bead i
2. Calculate acceleration of bead
3. Update bead velocity
4. Update bead position

How do we simulate polymer nanocomposites?

Molecular Dynamics Simulation

1. Polymer nanocomposites have very long relaxation times (must integrate Newton's Equations millions of times for each bead)

2. Most interesting polymer phenomena only occurs for large polymer lengths and system sizes (N becomes very large i.e. force calculation is expensive)

The force calculation in MD scales $O(N^2)$ where N is the number of atoms.

This is the dominant bottleneck in these simulations

For each bead in simulation…

1. Calculate force on bead i
2. Calculate acceleration of bead
3. Update bead velocity
4. Update bead position

How do we achieve the large system sizes and long time scales needed for polymer nanocomposite study?

Software Improvements

Neighbor Lists

Neighbor List Sorting

Memory Chunking

Domain Decomposition

Hardware Improvements

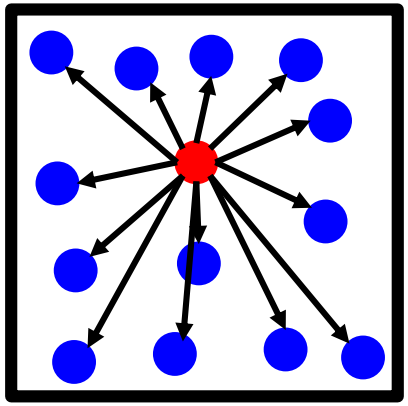
CPU Clock Speed

Memory Bandwidth

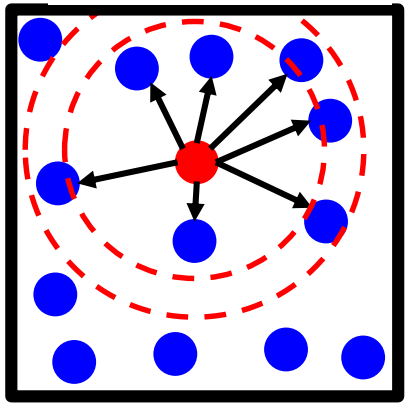
CPU Parallelization

GPU Parallelization

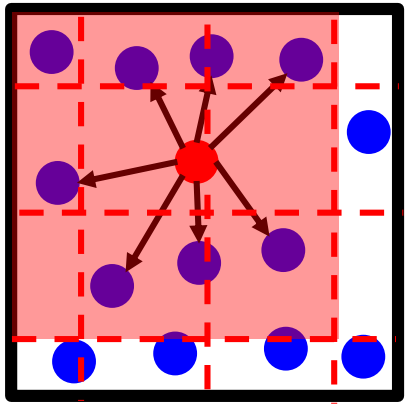
No Neighbor List



Verlet Lists

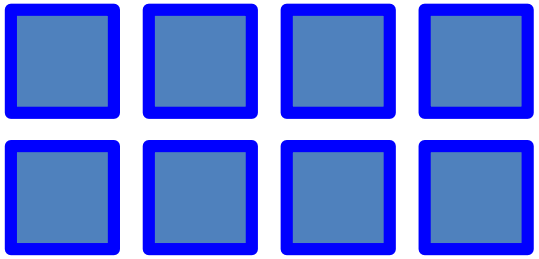


Cell List



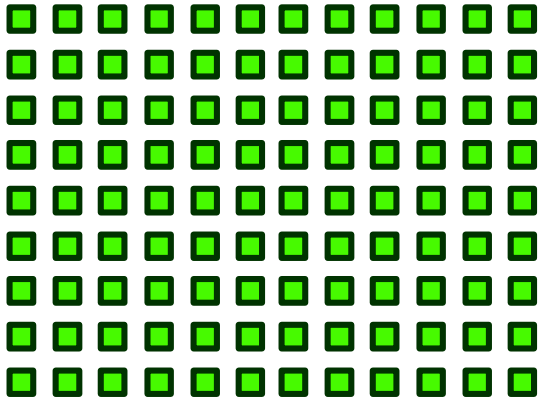
How do GPUs improve MD simulations?

CPU



A few high-speed processors

GPU



many low-speed streaming-multiprocessors

CPU's are designed to handle complex rapidly-varying tasks quickly

GPU's are designed to handle simple, highly-repetitive tasks very quickly

MD Simulation Steps

For each bead in simulation…

1. Calculate force on bead i
2. Calculate acceleration of bead
3. Update bead velocity
4. Update bead position

HOOMD-Blue: A General Purpose GPU-Based MD Simulator

https://codeblue.umich.edu/hoomd-blue/

```
from hoomd_script import *
init.read_xml('init.xml');
lj = pair.lj(r_cut=2.5)
lj.pair_coeff.set('A', 'A', epsilon=1.0, sigma=1.0)
integrate.mode_standard(dt=0.005)
integrate.nvt(group=group.all(), T=1.2, tau=0.5)
run(1e5)
```

```
$ hoomd run.py --mode=cpu
$ hoomd run.py --mode=gpu
$ mpirun -n 256 hoomd run.py --mode=cpu
$ mpirun -n 64 hoomd run.py --mode=gpu
```

Reasons to consider HOOMD-blue over other MD packages. HOOMD-blue…

- is purpose-built for the GPU
- is flexible enough to simulate simple bead-systems to complex bio-macromolecules
- leverages the full power of Python scripting

Journal of Computational Physics 230 (2011) 3556–3569

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Fast analysis of molecular dynamics trajectories with graphics processing units—Radial distribution function histogramming

Benjamin G. Levine [a,*,1], John E. Stone [b,1], Axel Kohlmeyer [a]

[a] Institute for Computational Molecular Science and Department of Chemistry, Temple University, Philadelphia, PA, United States
[b] Beckman Institute, University of Illinois at Urbana-Champaign, Urbana, IL, United States

ARTICLE INFO

Article history:
Received 15 September 2010
Received in revised form 22 January 2011
Accepted 31 January 2011
Available online 26 February 2011

Keywords:
Pair distribution function
Two-point correlation function
GPGPU

ABSTRACT

The calculation of radial distribution functions (RDFs) from molecular dynamics trajectory data is a common and computationally expensive analysis task. The rate limiting step in the calculation of the RDF is building a histogram of the distance between atom pairs in each trajectory frame. Here we present an implementation of this histogramming scheme for multiple graphics processing units (GPUs). The algorithm features a tiling scheme to maximize the reuse of data at the fastest levels of the GPU's memory hierarchy and dynamic load balancing to allow high performance on heterogeneous configurations of GPUs. Several versions of the RDF algorithm are presented, utilizing the specific hardware features found on different generations of GPUs. We take advantage of larger shared memory and atomic memory operations available on state-of-the-art GPUs to accelerate the code significantly. The use of atomic memory operations allows the fast, limited-capacity on-chip memory to be used much more efficiently, resulting in a fivefold increase in performance compared to the version of the algorithm without atomic operations. The ultimate version of the algorithm running in parallel on four NVIDIA GeForce GTX 480 (Fermi) GPUs was found to be 92 times faster than a multithreaded implementation running on an Intel Xeon 5550 CPU. On this multi-GPU hardware, the RDF between two selections of 1,000,000 atoms each can be calculated in 26.9 s per frame. The multi-GPU RDF algorithms described here are implemented in VMD, a widely used and freely available software package for molecular dynamics visualization and analysis.

HPC Symposium

Ahmadreza F. Ghobadi

Arthi Jayaraman's Research Group
Dep. of Chemical and Biomolecular Engineering &
Dep. of Material Science and Engineering
1/27/2016

What do I do with Farber?

Generate data

- Lammps
- Gromacs
- Gaussian
- Plumed

Analyze data

- C++
- Python
- VMD
- Plumed

Visualize data

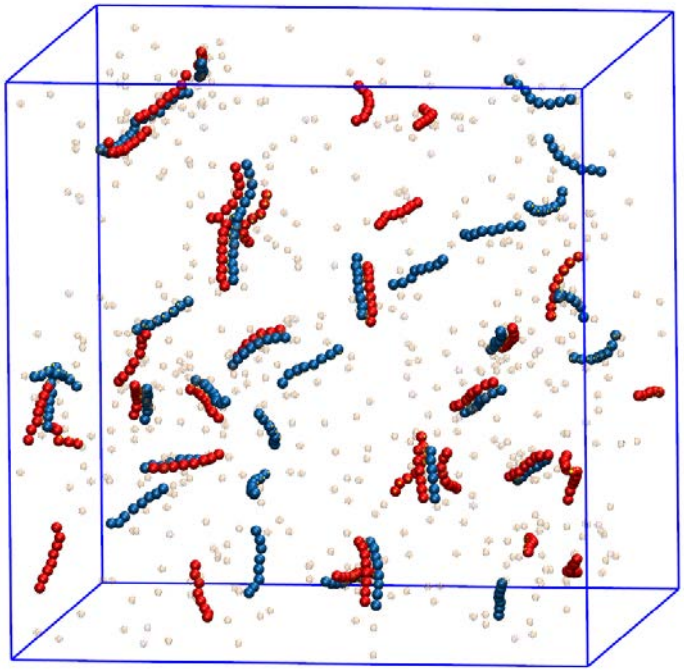
- GNUplot
- VMD
- OriginLab

Backup data

Generate Data (using Lammps)

Project definition:

- Relatively small system of interacting molecules
- Two different temperature (T1 & T2)
- Three simulation trials
- Each simulation needs 24 hrs on 10 cpu cores (no GPU)

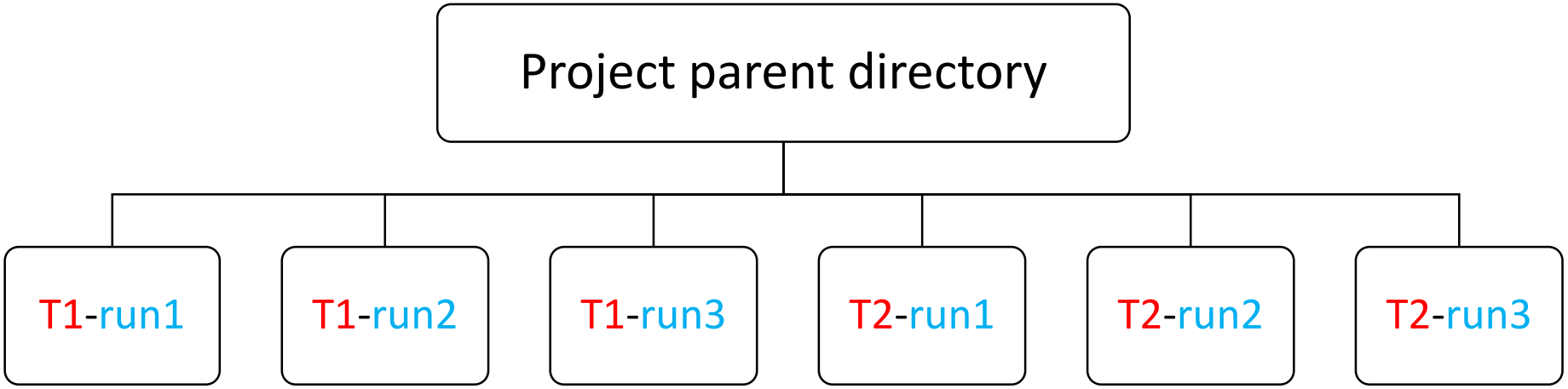


A.F. Ghobadi and A. Jayaraman, Soft Matter (2016, advance article), DOI: 10.1039/C5SM02868J

File preparation:

- Simulation input files (data file, etc)
- Simulation submission file (qs file)
- Scripts to submit multiple simulations at once

Generate Data (using Lammps) – Input files



Simulation input file (inp_eq):

```
Read initConf.dat
T = temp   #Temperature
Random number   = XXX
…
…
Write restart*
```

Simulation restart file (inp_res):

```
Read restart*
T = temp
Random number   = XXX
…
dump output to temp-runY-prodZ.dat
Write restart*
```

Generate Data (using Lammps) – Submission file

“submit.qs” file:

```
#$ -pe threads 10                  # export OMP_NUM_THREADS=$NSLOTS
#$ -l m_mem_free=2G                # Memory that must be available
#$ -l standby=1
# -l exclusive=1
#$ -l h_rt=8:00:00
# -m eas
# -M ahmadg@udel.edu
#$ -o dummy.o
#$ -e dummy.e
#$ -V                              # This is generally ignored
#$ -N dummy                        # Names the job to dummy
export VALET_PATH=$VALET_PATH: …some path…
vpkg_require jlab-lammps/ahmad
lmprun_FLAGS = ″input_file “
#WANT_CPU_AFFINITY=YES
MY_EXE = lmp
mpirun ${OPENMPI_FLAGS} $MY_EXE < $lmprun_FLAGS
```

Generate Data (using Lammps) – Bash script

“runs.sh” file:

```
#!/bin/bash
for T in T1 T2
    for run in 1 2 3
        … some lines …
        RND1=$RANDOM
        sed -i "s/temp/$T/g" inp_eq
        sed -i "s/XXX/$RND1/g" inp_eq
        … some lines …
        job = job_name_$T_$run
        sed -i "s/dummy/$job/g" submit.qs
        sed -i "s/input_file/inp_eq/g" submit.qs
        … some lines …
        jid = `qsub -terse submit.qs`
         … some lines …
        for next in 1 2 3
            … replace ZZ in inp_prod file with $next …
            jid = `qsub -terse -hold_jid $jid submit.qs`
        done
    done
done
```

Log into Farber

Navigate to your workgroup

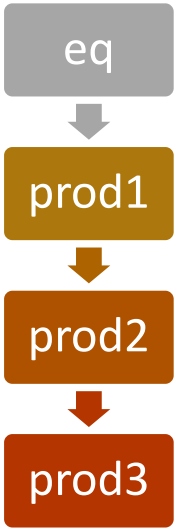
Navigate to $Lustre/scratch/…

[Parent directory] $ bash runs.sh

[Parent directory] $ qstat

some jobs …… r

some jobs …… hqw



Analyze Data – Serial Jobs

"analyze.sh" file:

```
#!/bin/bash
for T in T1 T2
    for run in 1 2 3
        … some lines …
        Read temp-runY-prod3.dat
        g++ analyze
        … some lines …
    done
done
```

"analyze.py" file:

```
Import os ??
for T in T1 T2
    for run in 1 2 3
        … some lines …
        Read temp-runY-prod3.dat
        python myscript.py (T1*a)/2
        … some lines …
    done
done
```

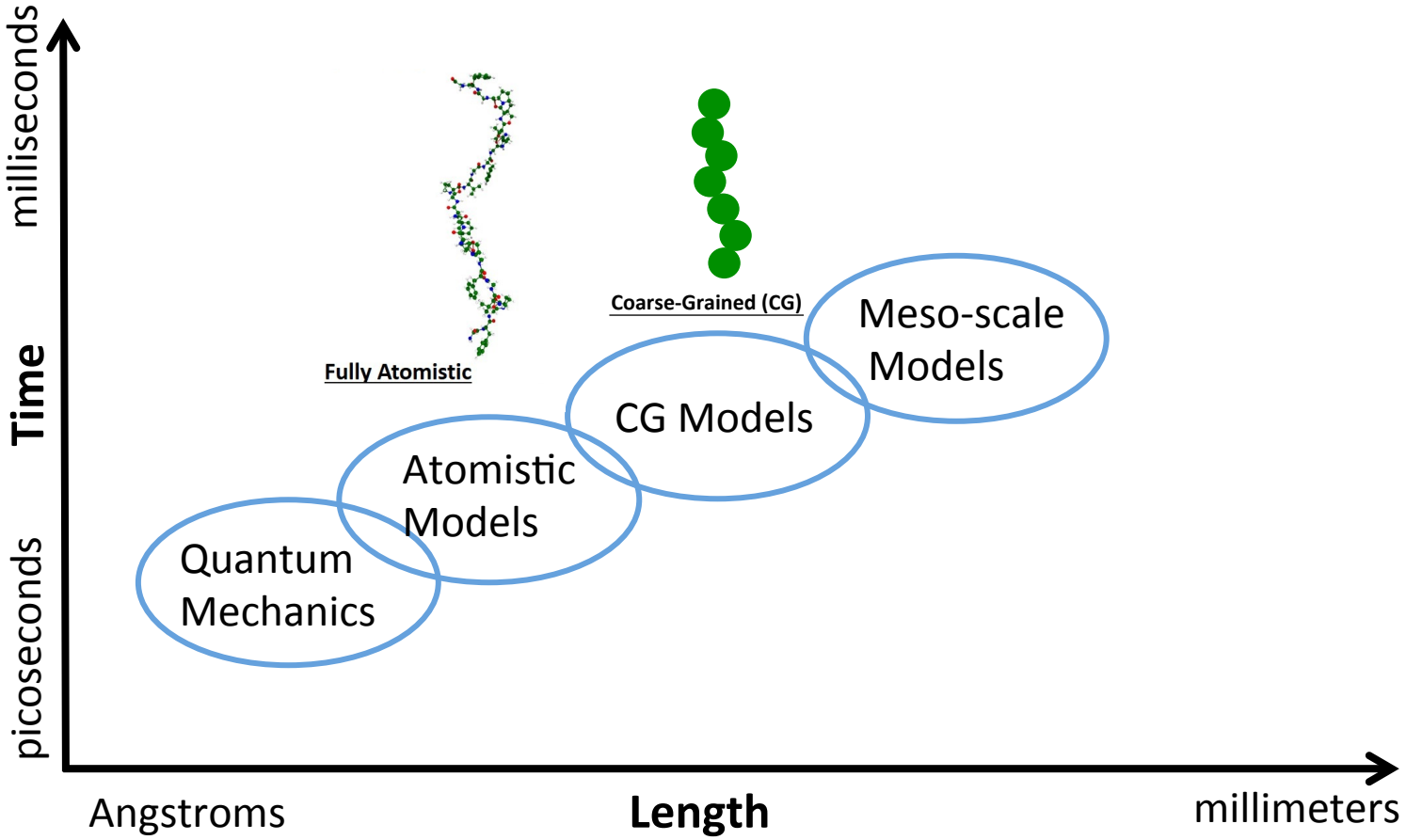
[Parent directory] $ qlogin (to ask 1 core from Jayaraman_lab.q)
[Parent directory] $ vpkg_require *some packages*
[Parent directory]@some_node $ bash analyze.sh
Or
[Parent directory] $ vpkg_require *python-numpy*
[Parent directory]@some_node $ python analyze.py

Using GROMACS and Open MPI to perform Parallel Atomistic Molecular Dynamic (MD) Simulations on Farber

Francesca Stanzione
Prof. Arthi Jayaraman's group

HPC Symposium 2016
January 27th 2016

Atomistic Molecular Dynamics

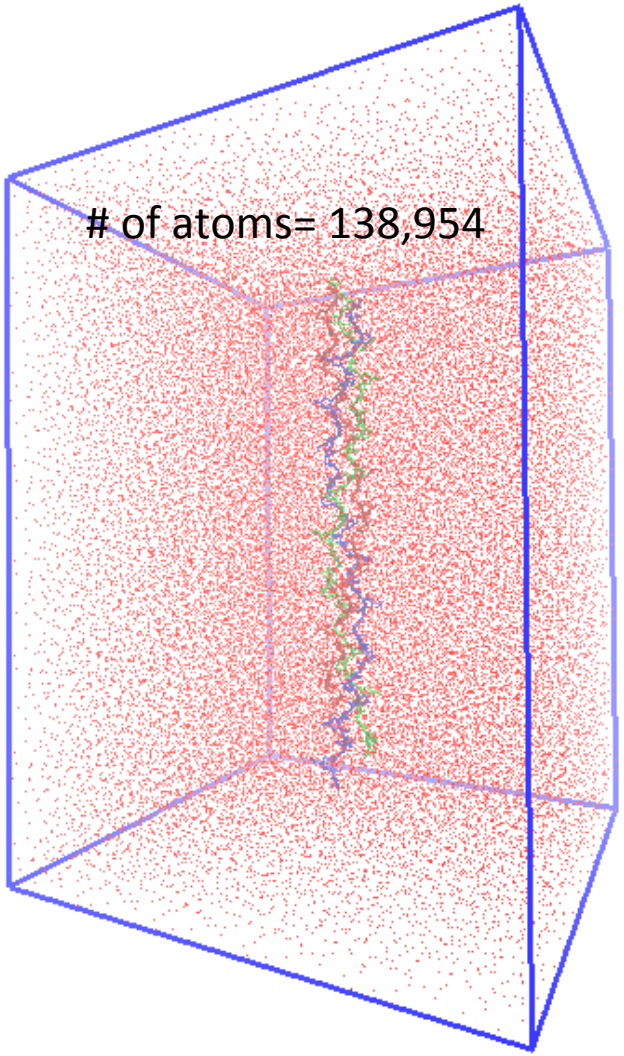


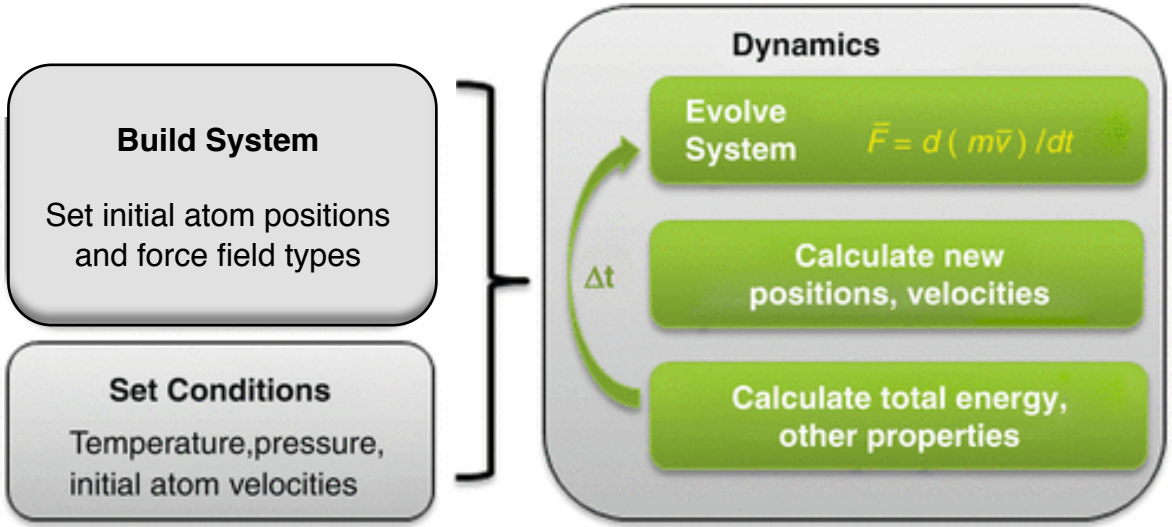
Several software for Atomistic Molecular Dynamics: GROMACS, Amber, NAMD, CHARMM…

GROMACS one of the fastest molecular dynamics packages and it is implemented on Farber

Atomistic MD Simulation on Farber

GROMACS available on Farber: Version 4.6.7 and version 5.0





Atomistic MD simulations are computationally more expensive than CG MD simulations and require more than 1 node

Parallel simulations on multiple nodes: Open MPI

Open MPI: the dominant multi-node parallelization-scheme

my_gromacs.qs

```
#$ -pe mpi  120
#$ -l m_mem_free=2G
#$ -q jayaraman_lab.q
#$ -l h_rt=04:00:00
#$ -l exclusive=1
#
#$ WANT_CPU_AFFINITY=YES
vpkg_require gromacs/double
MDRUN_FLAGS="-s run.tpr -deffnm run -npme -1"
# Which variant of mdrun?
#
if [ "x$USE_SINGLE_PRECISION" = "x" ]; then
  MY_EXE=mdrun_mpi_d
else
  MY_EXE=mdrun_mpi_d
Fi
.....
echo "GridEngine parameters:"
echo "  mpirun        = "`which mpirun`
echo "  nhosts        = $NHOSTS"
echo "  nproc         = $NSLOTS"
echo "  executable    = "`which $MY_EXE`" $MDRUN_FLAGS"
echo "  MPI flags     = $OPENMPI_FLAGS"
echo "-- begin Gromacs run --"
mpirun ${OPENMPI_FLAGS} $MY_EXE $MDRUN_FLAGS
echo "-- end Gromacs run --"
```

Parallel environment and number of processors desired

Our group nodes but it could be replaced by `-l standby=1`

With this flag I ask to use all 20 CPUs of the node

GROMACS specifics

Open MPI specifics (next slide)

Grid-engine & Open MPI

N.B. Detailed **gromacs.qs** file is available on Farber:

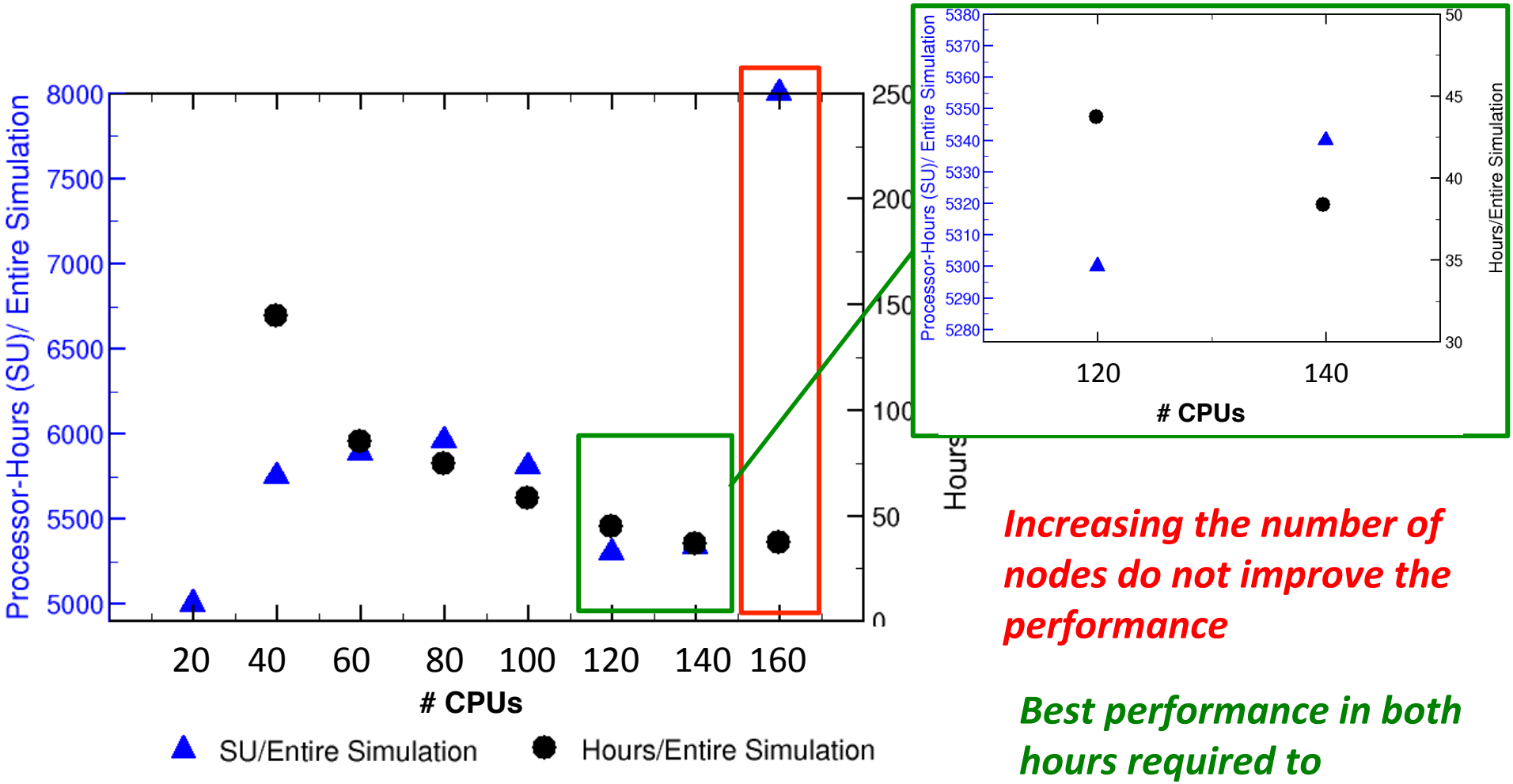
`/opt/templates/gridengine/gromacs`

Open MPI Specifics

```
OPENMPI_FLAGS="--display-map --mca btl ^tcp"
if [ "x$WANT_CPU_AFFINITY" = "xYES" ]; then
  OPENMPI_FLAGS="${OPENMPI_FLAGS} --bind-to core"
fi
if [ "${WANT_NPROC:-0}" -gt 0 ]; then
  OPENMPI_FLAGS="${OPENMPI_FLAGS} --np ${WANT_NPROC} --map-by node"
fi
if [ "x$SHOW_MPI_DEBUGGING" = "xYES" ]; then
  OPENMPI_FLAGS="${OPENMPI_FLAGS} --debug-devel --debug-daemons --
display-devel-map --display-devel-allocation --mca mca_verbose 1 --
mca coll_base_verbose 1
--mca ras_base_verbose 1 --mca ras_gridengine_debug 1 --mca
ras_gridengine_verbose 1 --mca btl_base_verbose 1 --mca
mtl_base_verbose 1 --mca plm_base_verbose
 1 --mca pls_rsh_debug 1"
  if [ "x$WANT_CPU_AFFINITY" = "xYES" -o "x$WANT_HALF_CORES_ONLY" =
"xYES" ]; then
    OPENMPI_FLAGS="${OPENMPI_FLAGS} --report-bindings"
  fi
fi
```

Benchmarking

to evaluate how many resources are required by a single simulation



SU= hour of computing on 1CPU

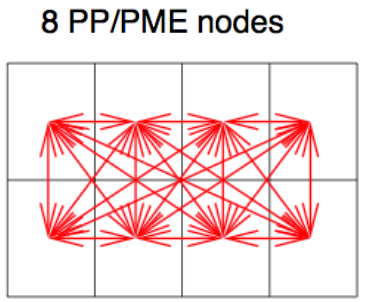
Hours required to complete a single simulation

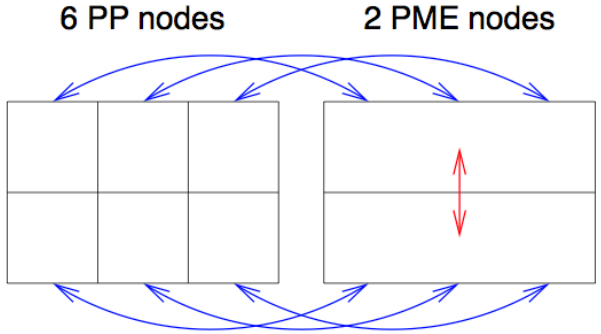
Increasing the number of nodes do not improve the performance

Best performance in both hours required to complete the simulation and hours of computing

Optimizing the Performance with GROMACS

A crucial calculation to be optimized is the long range interactions which consist of dispersion-type particle-particle (PP) interactions and the electrostatic forces obtained from the Particle Mesh Ewald (PME) method.





Particle communications

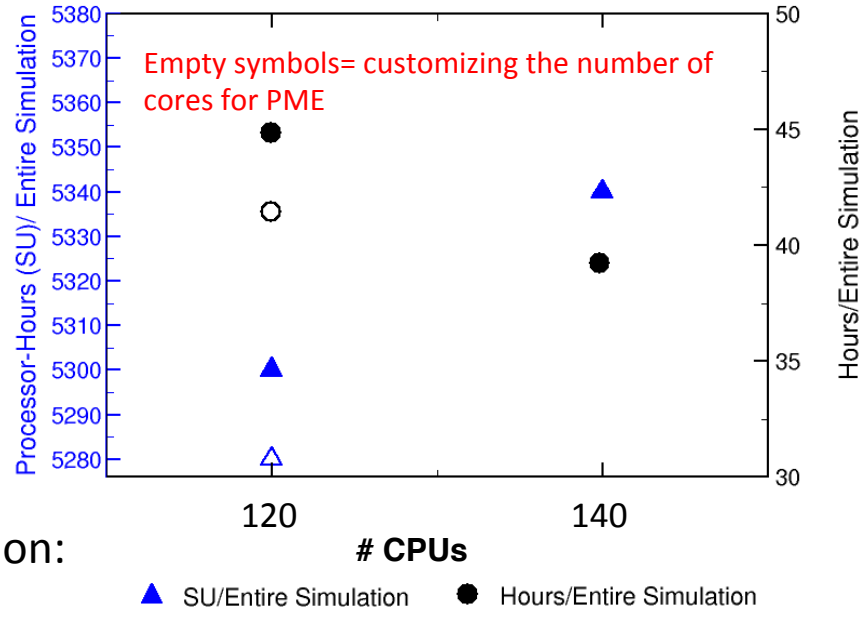
PP - PME coordinate and force communication

M.J. Abraham, D. van der Spoel, E. Lindahl, B. Hess, and the GROMACS development team, GROMACS User Manual version 4.6.7, www.gromacs.org (2014)

my_gromacs.qs

```
#$ -pe mpi  120
#$ -l m_mem_free=2G
#$ -q jayaraman_lab.q
#$ -l h_rt=04:00:00
#$ -V
#$ -l exclusive=1
vpkg_require gromacs/double
#USE_SINGLE_PRECISION=NO

MDRUN_FLAGS="-s run.tpr -deffnm run -npme -1"
```



To set the number of cores only for PME calculation:

```
MDRUN_FLAGS="-s run.tpr -deffnm run -npme 40"
```

Optimizing the Performance with Grid-Engine flags

my_gromacs.qs

```
#$ -pe mpi  128
#$ -l m_mem_free=2G
#$ -q jayaraman_lab.q
#$ -l h_rt=04:00:00
#  -l exclusive=1
#$ -R y
vpkg_require gromacs/double
$ WANT_CPU_AFFINITY=YES

MDRUN_FLAGS="-s run.tpr -deffnm run -npme -1"
```

Disable the exclusive flag

Reserve the minimum number of nodes to satisfy my CPUs request